DEVELOPMENT OF AN OPTIMIZATION COMPONENT FOR THE SOUTH FLORIDA REGIONAL SIMULATION MODEL

A Thesis

Presented to the Faculty of the Graduate School

of Cornell University

In Partial Fulfillment of the Requirements for the Degree of

Master of Science

by

Christopher Daniel Andrew Boone

January 2009

ABSTRACT

South Florida has a large number of water bodies, including rivers, lakes, and canals. Many of these water bodies are separated by manmade structures that allow water managers to control the flow throughout the region. These water managers are tasked with meeting a variety of objectives related to flood control, water supply, recreation, and the well-being of the ecosystem.

Computer models are used to assist in meeting these objectives. In particular, the South Florida Water Management District employs simulation models that aim to describe the flow of water throughout the region. Simulation models allow managers to use computers to test the effects of changes to the system. In South Florida, the flow of water is governed by the hydrological characteristics of the region, as well as the actions of the individuals operating the flow control structures. In order to produce an accurate representation of the system, the simulation models in South Florida must incorporate both sets of characteristics.

For this project, a Linear Programming (LP) model has been developed in cooperation with the South Florida Water Management District (SFWMD). The model contains the major water bodies and connecting structures in South Florida. The purpose of this program is to model the actions of the operators of the flow control structures. Given targets on the water levels in the various water bodies, the LP model allocates flows in a way that most closely meets the targets, subject to a variety of constraints. These constraints include physical characteristics of the system, as well as constraints imposed by the water managers. As an optimization model, our contention is that the

LP model can be used to simulate the actions of optimizing agents (i.e., the water managers in South Florida).

We compare the results of the LP model to those from the Object Routing Model (ORM), a computer program that has been developed and used by the SFWMD for the same purpose. The purpose of the comparison is to determine if the LP model can produce results similar to those of ORM, thereby giving support to the possibility of incorporating a linear programming component within the simulation model currently under development at the SFWMD.

BIOGRAPHICAL SKETCH

Christopher Daniel Andrew Boone was born in Rochester, New York. He graduated from Cornell University in 2005, earning a B.A. in Philosophy and a B.S. in Civil and Environmental Engineering. Christopher remained at Cornell University for graduate studies in the School of Civil and Environmental Engineering. Upon completing his Master's degree at Cornell, Christopher enrolled in a Ph.D. program in Economics at Columbia University.

ACKNOWLEDGEMENTS

I would like to thank the South Florida Water Management District for its cooperation in this research. In particular, I owe my gratitude to the staff at the Office of Hydrologic and Environmental Systems Modeling, including Jayantha Obeysekera, Randy Van Zee, and Joseph Park.

I also owe thanks to José Aponte and Dr. Jonghwa Ham, my colleagues at Cornell with whom I collaborated on this research. Both of these individuals contributed significantly to the development of the model, including conceptual development and coding.

I would also like to thank the faculty and staff of the School of Civil and Environmental Engineering, especially James Bisogni and Monroe Weber-Shirk.

Finally, I owe special thanks to the members of my committee. I am grateful to Harry de Gorter for serving as my minor advisor, and for his guidance and willingness to discuss my research interests. I owe my greatest thanks to my advisor, Daniel P. Loucks, for supervising this research and offering useful feedback, and for his continuous support, patience, and advice throughout the course of my graduate studies.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

GLPK	GNU Linear Programming Kit
HSE	Hydrologic Simulation Engine
LP	Linear Programming, or Linear Program
MSE	Management Simulation Engine
ORM	Object Routing Model, or Object-oriented Routing Model
RSM	Regional Simulation Model
SFWMD	South Florida Water Management District

CHAPTER 1

INTRODUCTION

Water management in South Florida is a complex task due to the unique characteristics of the region, which include thousands of kilometers of canals and levees, heavily populated urban areas, large bodies of water like Lake Okeechobee and the water conservation areas, and ecologically important areas like the Everglades.[1] The region contains hundreds of manmade flow control structures, such as gates and pumping stations, that allow water managers to control the flow between various water bodies. These operators must make decisions about how best to manage water in order to meet a variety of different objectives, including flood management, municipal and agricultural water supply, water quality, and environmental restoration. The South Florida Water Management District (SFWMD) is the government agency tasked with managing the water to meet these objectives.

Years of development in South Florida have altered the historical ("natural") flow of water through the region, and this has had significant impacts on the ecosystem. A major goal of the restoration efforts is to manage the flow of water in a way that sustains and improves the viability and diversity of the ecosystem. SFWMD is the lead agency involved in the Comprehensive Everglades Restoration Program (CERP), "the largest environmental project in the nation's history."[2] CERP provides "a framework and guide to restore, protect and preserve the water resources of central and southern Florida, including the everglades…. [It] will take more than 30 years to construct and will cost an estimated $7.8 billion."[3] The main focus will be to capture

[1] Lal *et al*. (2005); http://www.sfwmd.gov (accessed 11/23/2008).
[2] http://www.sfwmd.gov (accessed 11/23/2008).
[3] http://www.evergladesplan.org/about/about_cerp_brief.aspx (accessed 11/23/2008).

additional fresh water before it flows to the ocean, and allocate this water to revive the dying ecosystem.[4]

Simulation models are used to assist in water resources planning and management. By allowing water managers to predict the hydrological and ecological impacts of alternative water management strategies or policies under various sets of conditions, simulation models can help them identify which policies better meet various planning and management objectives. As part of the Everglades restoration project in South Florida, for example, there are a large number of possible courses of action, and it becomes necessary to compare alternatives in order to choose the best ones. To aid in this task, SFWMD has been developing the Regional Simulation Model, or RSM, an object-oriented simulation model for use in water resources planning.[5] A detailed simulation model like RSM can give planners insight on how proposed policies, such as new infrastructure projects or changes in management procedures, might affect the system and how well it satisfies various hydrological and ecological targets.

This thesis describes work performed for the Office of Hydrologic Systems Modeling at the South Florida Water Management District. The main objective of this work was to create a linear optimization model to describe the flow of water throughout South Florida. In doing so, we were trying to determine if this model can produce results that are similar to those produced by an RSM implementation of the same system, called the Object Routing Model (ORM).[6]

[4] http://www.evergladesplan.org/about/about_cerp_brief.aspx (accessed 11/23/2008).
[5] SFWMD (2005d).
[6] See Park *et al*. (2007). The ORM is also known as the Object-oriented Routing Model. The version that we are working with represents the *sfBasins* network from August 2006.

The ORM is a basin routing model that includes the major basins and flow control structures in South Florida. The basins represented include: Lake Okeechobee; the Lake Okeechobee service areas; the major canals and rivers, including St. Lucie, Caloosahatchee, West Palm Beach, Hillsboro, North New River, and Miami; the water conservation areas; Everglades National Park and Big Cypress National Preserve; and the lower east coast service areas. The optimization model that we have developed models this same network, and we compare our results to the ORM results. It is a node-link optimization model, with the nodes representing the basins, and the links representing the flow control structures. A diagram of this network is given in Figure 1.

In real life, the flow of water throughout South Florida (given hydrological inputs) is governed by: (1) the hydrological and hydraulic characteristics of the physical system, including the natural landscape and manmade infrastructure, and (2) the management decisions made by the operators of flow control structures. In order to accurately model the water flow, these models must take into account both of these factors.

In order to model the management decisions at flow control structures, RSM (and, therefore, ORM) uses a rule-based approach. The program contains objects that determine the flow at each structure by following a series of rules; these rules must be explicitly defined in the code. The optimization model, on the other hand, uses a system-wide linear programming (LP) solver to determine the set of flows that allows the system to best meet the desired targets. The targets have associated priorities ("weights") that characterize the tradeoffs between meeting different sets of targets or management objectives.

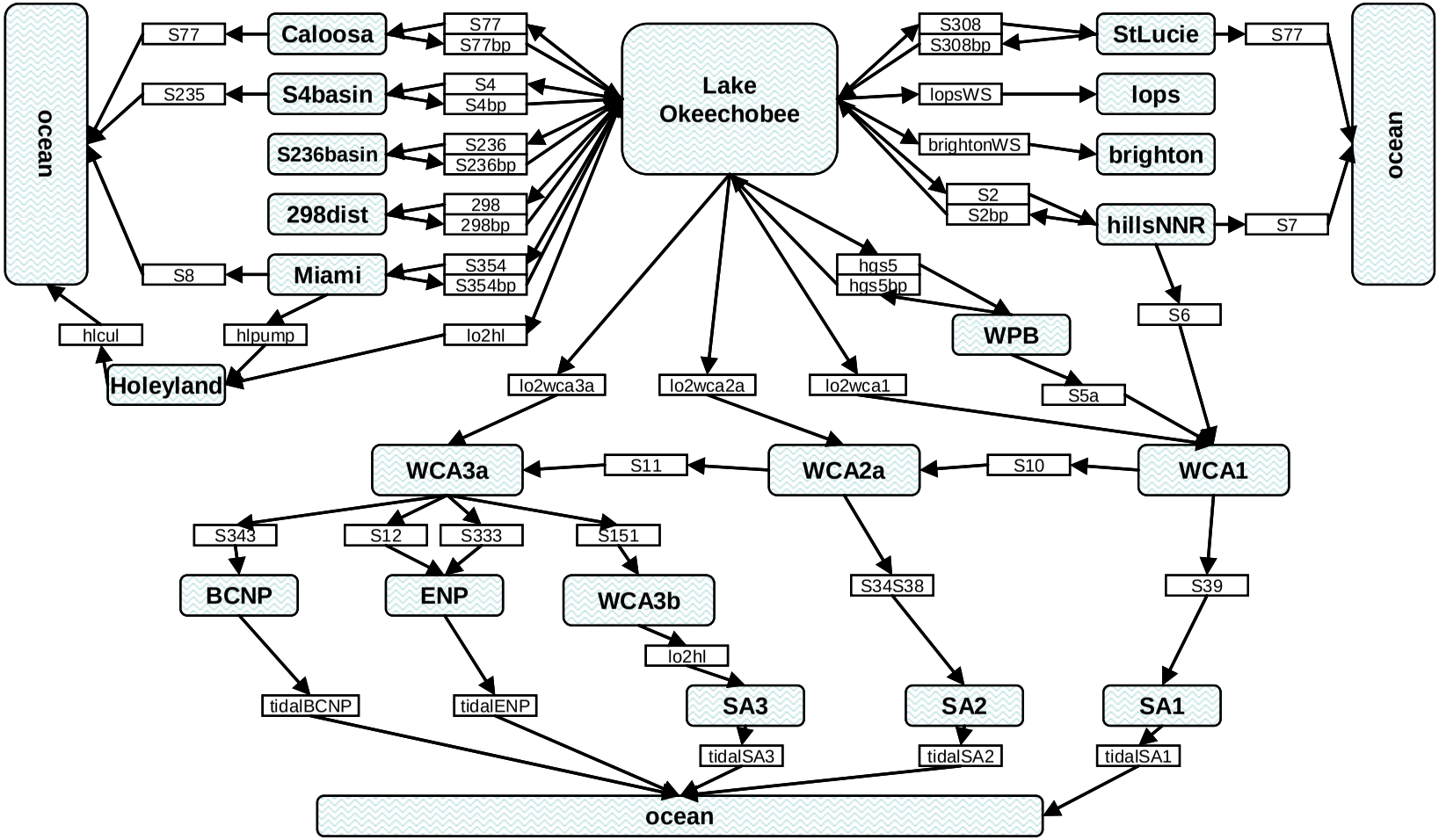


Figure 1 Diagram of the network modeled by the ORM and by the optimization model described in this report.

The LP model described here is being developed with the purpose of eventually integrating it into the RSM to assist with modeling the management decisions. This report discusses the prospects for such an outcome. The desire for an LP component to the RSM stems from difficulties in modeling management decisions over large or complex networks. While the ORM is able to adequately model its relatively simple network, it can become difficult to explicitly define sets of rules to govern flows in larger, more complex networks. By creating an LP model for the ORM network and examining the results, we can evaluate the potential for integrating the LP model into the RSM.

The goals of this research included finding answers to the following questions:

- By assigning different priorities ("weights") to the targets on each basin, how well can the LP model mimic the management decisions made by the ORM?
- Can these results be improved by explicitly implementing some of the rules used in the ORM?
- What are the major difficulties associated with creating and implementing such an LP model?

We conclude that the LP optimization approach shows promising results, and should be explored further as an alternative for modeling water management decisions throughout the South Florida region. The results of our analysis, along with a discussion of the LP model's strengths and shortcomings, as well as guidelines for future work, are given in the following sections.

CHAPTER 2

THE LINEAR PROGRAMMING MODEL

2.1. Overview of Linear Programming

Consider the following constrained optimization program:

$$
\begin{array}{ll}
\text{minimize} & Z(x_1,\ldots,x_n) \\
\text{subject to} & g_1(x_1,\ldots,x_n) = b_1 \\
 & \vdots \qquad\qquad \vdots \\
 & g_k(x_1,\ldots,x_n) = b_k \\
 & \\
 & h_1(x_1,\ldots,x_n) \le c_1 \\
 & \vdots \qquad\qquad \vdots \\
 & h_m(x_1,\ldots,x_n) \le c_m
\end{array}
$$

The function whose value is to be minimized, $Z(x_1,\ldots,x_n)$, is called the *objective function*. The other functions are the *constraints*. The variables, x_1, x_2,…, x_n are the *decision variables*. A linear program is simply an optimization program like the one shown above in which the objective function and all constraints are linear. A linear program can therefore be written in the following form[7]:

$$
\begin{array}{ll}
\text{minimize} & \mathbf{c'x} + d \\
\text{subject to} & \mathbf{Ax} \le \mathbf{a} \\
 & \mathbf{Bx} = \mathbf{b}
\end{array}
$$

Linear programming has a number of attributes that make it an attractive approach to solving water resources problems.[8] One advantage is that any solution that is found to the constrained optimization problem will be a global solution (though it is not necessarily unique). The problem of local (non-global) optima that can afflict

[7] Using vector notation **x** is an n-by-1 column vector of decision variables, **c'** is a 1-by-n row vector of coefficients, d is a scalar, **A** is a k-by-n vector of coefficients, **B** is an m-by-n vector of coefficients, and **a** and **b** are k-by-1 and m-by-1 column vectors of coefficients, respectively. A linear program may include additional constraints stipulating that the decision variables must be nonnegative. For further information about linear programming, see Dantzig (1963) and Luenberger and Ye (2008).

[8] For more information about the use of optimization and linear programming in water resources problems, see Loucks and van Beek (2005).

nonlinear problems does not exist with linear problems. Another advantage of linear programming is that there exist efficient computer-based solvers that are able to find solutions very quickly. These two attributes make linear programming useful for solving problems containing very large numbers of variables and constraints.

2.2. Overview of RSM and ORM

RSM is being developed by the SFWMD in order to address the specific complexities involved in modeling the South Florida region. RSM is designed to model both groundwater and surface water – and their interaction. It is a physically-based model which divides the region into finite volume elements and keeps track of the water contained in these elements and the flow between adjacent elements. RSM uses object-oriented code design. It integrates the modeling of hydrologic characteristics with the modeling of management decisions.[9]

In its applications for South Florida, RSM is meant to be used as a planning model. That is, it is not meant to determine what decisions to make at flow control structures during day-to-day operations; instead, RSM is to be used to compare alternative management policies in order to determine those that best satisfy competing objectives, such as meeting water supply and flood control needs.

RSM consists of two coupled components: the Hydrologic Simulation Engine (HSE) and the Management Simulation Engine (MSE). The HSE simulates the hydrologic characteristics, solving the governing equations for flow through the natural system as well as through man-made structures. The MSE simulates the management capabilities of the flow control structures. The HSE provides the MSE with basic

[9] For more information about RSM, see Lal *et al*. (2005), Park *et al*. (2007), and SFWMD (2005a-d).

hydrologic information about the state of the system. The MSE uses this information to make management decisions. Then the HSE continues the simulation given these operational decisions.[10]

The MSE consists of a multilayer, hierarchical management structure, consisting of multiple classes of management objects. For each managed flow control structure there is an associated *controller*, which limits the flow through that structure according to management objectives. Above the controllers in the hierarchy is a layer of *supervisors*, which manage the controllers and coordinate the global behavior of the flow control structures. There are also *assessors* which process information about the variables important to the management decisions and provide this information to the supervisors. Each of these components of the MSE is integrated with the hydrological computations of the HSE.[11]

The ORM represents a simplified version of the RSM, and is specific to the South Florida region. Only the major basins and structures are represented, as described above. The basins are represented as "waterbody" objects and the structures as "watermovers." There are no groundwater-surface water interactions; instead, there is only one type of flow, and in only one dimension. A specialized RSM supervisor module was created for implementing the ORM. Many of the management decisions governing structure flows are coded in the form of binary decision trees. Assessors are used to quantify water supply and flood control needs and determine the final structure flows.[12]

---

[10] SFWMD (2005d).
[11] Park *et al.* (2007)
[12] See Park *et al.* (2007) for information about MSE and ORM. ORM is essentially an object-oriented implementation – based on RSM – of the earlier South Florida Regional Routing Model (SFRRM); for information about SFRRM, see Trimble (1986).

The binary decision trees in the ORM are implemented in objects called *coordinators*. While the assessors are contained in the source code, the coordinators are defined via XML input files, allowing for relative ease in making adjustments to the management constraints. Each coordinator (decision tree) is specific to only one flow control structure. In addition, each coordinator regulates the flow for only one purpose, either water supply or flood control. For example, the structure regulating releases from Lake Okeechobee to Water Conservation Area 1 has two coordinators associated with it, one regulating the flood control releases and the other regulating water supply releases. Structure S77, which regulates releases from Lake Okeechobee into the Caloosahatchee River, has only one coordinator, regulating the flood releases. Some of the structures in the network have no coordinator associated with them.

In the ORM, the system is analyzed separately for the purposes of flood control and meeting water supply demands. The assessors are responsible for determining the final structure flows that take into account any tradeoffs between these two objectives. While the coordinators provide information about how the flows should be constrained in order to meet one of these objectives, the constraints contained within the coordinators cannot necessarily be interpreted as constraints on the total flow. And while the assessors follow an explicit set of rules to determine the structure flows, they are contained within the ORM source code, and there is, unfortunately, little documentation discussing these methods. These obstacles make it difficult to fully analyze the methods that the ORM uses to determine its flows.

2.3. Details of the LP model

At the request of the SFWMD, the linear programming solver used in our work is the GNU Linear Programming Kit (GLPK).[13] GLPK is part of the GNU project and is released as "free software" under the GNU General Public License.[14] GLPK requires no licensing fees, and the software can be made freely available to anyone wishing to use the RSM.

While the goal of constructing the LP model is to integrate it with RSM, the model described in this report runs independently of RSM. In order to make this possible, we made use of an additional software application: MATLAB.[15] GLPK is used to solve the LP optimization model for each time step, while MATLAB is used for everything else: pre-processing the input data and creating the GLPK input files for each time step; calling the GLPK solver; post-processing the LP results; and displaying the output. If the LP model is to be integrated into RSM, MATLAB would not be necessary, as this functionality would need to be added to RSM. The LP model would remain, and RSM would directly call the GLPK solver for each time step.

In the LP model, each basin in the network is assigned a target storage value, as well as a set of weights that represent the relative penalties to be applied to deviations from this target value. Different weights can be applied to deviations above the target (excess) and deviations below the target (deficit). Basins in our model can also have multiple storage targets, with different weights for each target. The deviation from each target is multiplied by its respective weight, and this product is summed over all

[13] See http://www.gnu.org/software/glpk/ for more information.
[14] See http://www.gnu.org/copyleft/gpl.html for more information.
[15] MATLAB is a high-level programming language and an interactive numerical computing environment. For information about MATLAB, see http://www.mathworks.com/products/matlab/.

targets and basins. This final weighted sum represents the objective function that is to be minimized for the LP model.

The major decision variables in the LP model are the flows between the basins. The LP solver adjusts these flows in order to find the minimum value of the objective function.

The final version of the LP model optimizes over the entire region for a single time step – generally one day. The inputs to the model include the following for each basin: initial stage, target stage, rainfall, runoff, and potential evapotranspiration. The model then produces the following output: flow through each structure during the time period, and final stage in each basin.

A separate GLPK input file is created by MATLAB for each time step. The GLPK solver produces an output file with the results of the optimization. These results are read by MATLAB, which can then be used to display or analyze the results. If the system is being analyzed over multiple time periods, MATLAB can use the LP results from the previous time period to create an input file for the next time period. This allows us to run the overall model for multiple days in succession, using the stage results from the LP at the end of one time period as the initial stages for the next time period. This provides a convenient way to be able to compare the results of the LP model to the results of RSM, since RSM is able to analyze a large number of time periods.

Our LP model optimizes over space, but not over time. There are a number of barriers to creating an LP model that can optimize over multiple time periods at once. Linear

programming requires that the objective function and all constraints be linear. However, a number of the constraints in the ORM are nonlinear, including many of the constraints governing structure flow capacities or management decisions. One way to overcome this difficulty could be to use linear approximations of the nonlinear constraints. However, this can reduce the accuracy of the model. Binary or integer variables would likely be required in order to reasonably approximate many of the constraints. While GLPK is able to solve mixed-integer programming models, there is a limitation on the number of binary variables that can be included. The runtime increases significantly with the inclusion of just a handful of binary variables, and the model becomes unsolvable after just a few hundred.

2.3.1. Objective Function

As noted above, each basin can actually have multiple storage targets. Most of the basins in our LP model have two targets, an upper target and a lower target. In general, for the upper target, a penalty is applied for any deviation above or below this value; for the lower target, only deviations below this value are penalized. Therefore, if the storage level in a basin is equal to the upper target level, the sum of weighted deviations for that basin equals zero. The addition of the lower target allows for a larger penalty to be applied when the water level in the basin falls below some critical level. Some of the targets remain constant no matter what time of year is being modeled; others vary throughout the year according to a rule curve. The objective function seeks to minimize the following weighted sum of upper target excesses, *target1Excess(i)*, and upper and lower target deficits, *target1Deficit(i)* and *target2Deficit(i)*, over all basins *i*:

$$\sum_i \; [\; weight\_t1E(i)*target1Excess(i) + weight\_t1D(i)*target1Deficit(i) + weight\_t2D(i)*target2Deficit(i) \;]. \qquad (1)$$

In general, the upper target corresponds to the *maintenance level* in the ORM network, and the lower target corresponds to the *reserve level*. Because the two models (LP and ORM) differ markedly in their structure, it is not necessarily the case that the maintenance and reserve levels from the ORM represent the targets that will lead to the best results in the LP model (i.e., the results that most closely match the ORM output).[16] Using the maintenance and reserve targets as the targets in the LP is a natural first step in creating the LP model, and is useful for comparing the results, but this is not meant to preclude the possibility of using different target values. This topic is discussed further in the next chapter.

Once the targets have been chosen, it is necessary to choose the weights that are used to penalize deviations from the targets. These weights allow for assigning different priorities to deviations from different basins. For example, suppose it is very important that basin A remain at or near its target level, while basin B may be allowed to fluctuate considerably without significant consequences. In such a case, a larger weight on deviations for basin A would reflect this relative importance. The weights also allow for adjustments to compensate for differences in the relative ease with which the model can maintain the water levels in different basins. For example, the characteristics of the system could make it such that it is generally easier to maintain the water level in basin A than in basin B. In this case, equal weights on deviations for both basins would result in larger deviations for basin B. Increasing the weight for

[16] Presumably, the best targets to feed to the optimization model – in order to match the ORM results – are the ORM results themselves. This would obviously do little to serve our present purpose, however.

basin B could force the LP solver to reduce the deviations for that basin in favor of deviations elsewhere.

The weights that are used to penalize deviations from the target levels were chosen by trial and error. Once the targets were selected for the LP model, the weights were adjusted until the LP stage results closely matched the ORM results over the 5-year time period for which ORM data were available. While an effort was made to find weights that would allow the LP results to closely match the ORM results, the final set of weights used in this thesis are not necessarily the best. The process of finding weights is one of the more difficult tasks associated with this sort of optimization model; it is not always clear what set of weights would be best, nor is it always clear what procedure should be followed in order to determine appropriate weights. It is not necessarily the case that the same trial and error procedure used to find the weights for this thesis would be most appropriate for determining the weights in a more complete LP model.

2.3.2. Constraints

The model incorporates a number of different constraints. One category of constraints represents the maximum daily flow capacity for each structure. These capacity constraints constitute the maximum amount of water that can physically flow through the structure in a single day. For some structures, the flow capacity is a constant value. For others, it can depend on a number of other variables, such as the water levels in the upstream and downstream basins or the flow in the previous time period. The equations for flow capacity for these structures are taken from the source code of the ORM.

Another set of constraints represents management constraints imposed as the result of regulations or operator decisions. An example of this type of constraint would be the bounds on flood releases from Lake Okeechobee into the Caloosahatchee and St. Lucie Rivers. These releases depend on the water level in Lake Okeechobee as well as the time of year (i.e., the rule curves). All of the management constraints implemented in the LP come from the coordinators in the ORM, where they are found in the XML input files. We were unable to implement all of the constraints contained in the ORM coordinators, since these constraints are specific to a particular flow purpose (water supply or flood control), and the LP model makes no distinction between different types of flow. By examining the ORM results, we could determine which of the management constraints appeared to be binding for total flow. Only these constraints were implemented; they correspond to the coordinators for structures S77, S308, S77bp, S308bp, hgs5bp, S10, and S343.

We also implemented minimum flow constraints for the three flowways between Lake Okeechobee and the water conservation areas (lo2wca1, lo2wca2a, lo2wca3a). A time series of minimum flow values is input into the ORM, and we use this same time series as minimum flow values for the LP model.[17]

Since some of the flow capacity and management constraints are nonlinear functions of other variables, the value of these constraints are pre-processed in MATLAB. Only the actual numerical value of the constraint for a particular time period is transmitted

[17] A note on terminology: While the minimum flow requirements might also be considered "management constraints," we make the distinction here because the minimum flow constraints are implemented differently from the management constraints in both the ORM and the LP model. The management constraints referred to above are implemented as *coordinators* in the ORM, while the minimum flow constraints are implemented as *mse_node* objects in the *mse_network*.

to the LP model. The flow capacity and management constraints are implemented in the LP in the following manner for each flow control structure (i,j):

$$flow(i,j) \leq constraintValue. \quad (2)$$

The minimum flow constraints are implemented as follows:

$$flow(i,j) \geq constraintValue. \quad (3)$$

Another important constraint maintains conservation of mass within the system. This sort of continuity constraint is common to many water resources optimization models. The continuity constraint from our LP model is shown in Equation (4).

$$\begin{aligned} FinalStorage(i) = {} & InitialStorage(i) + Rainfall(i) \\ & - Evapotranspiration(i) + Runoff(i) - WaterSupplyDemand(i) \\ & + DeltaStorage(i) + \sum_j flow(j,i) - \sum_j flow(i,j) \end{aligned} \quad (4)$$

for all basins *i* and *j*, where *FinalStorage(i)* represents the storage in basin *i* at the end of the time period, and *InitialStorage(i)* represents the storage at the beginning of the time period.[18] The LP model determines the flows between the basins, and these flows establish the final storage value for each basin according to Equation (4).

[18] The *DeltaStorage* values are input to both the ORM and LP model; this variable represents any additional volume adjustments that must be made for that time period. See Trimble (1986) for more information.

CHAPTER 3

RESULTS, DISCUSSION, AND FUTURE WORK

3.1. Results

This section describes the output from the LP model, and compares this output to that of the ORM. We compare results for stage levels in the basins and for flows between the basins. Daily ORM output was available for the 5-year period between January 1, 1965 and December 31, 1969. In order to compare the performance of the LP model with that of the ORM, the LP model was run for the same 5-year time period, using the same input data that were used in the ORM. Each model begins with the same initial stages in the beginning of the first time period. In subsequent time periods, the LP model receives no information about the ORM stages; instead, the initial stage in each basin is set equal to the final stage in the previous time period.

The results from multiple versions of the LP model, which differ according to the constraints that were imposed, are discussed in this section. The first version is the full LP model. It contains management constraints and minimum flow requirements for some of the structures, along with maximum flow (capacity) constraints for all structures. For the second version, we run a simplified LP model where we exclude all of these extra flow constraints. The third version includes maximum flow constraints, but excludes the management constraints and minimum flow requirements. Examining multiple sets of results allows us to assess the impacts of the various model components.[19]

[19] The same set of weights was used for each of the three versions.

3.1.1. Version 1: Includes all flow constraints

The first set of results that we present is from the complete model containing all of the flow constraints mentioned above: flow capacity constraints, minimum flow requirements, and management constraints. The basin stage results for this run of the LP are shown in Figure 2. The ORM stage results and the target levels are shown in the figure as well. The results shown here are for selected basins; for the full set of results, see Appendix A.

The weights used to produce these results represent the best set of weights that we found. The results can differ significantly depending on the weights chosen. See APPENDIX B for an example of the results under a slightly different set of weights. A comparison between these two sets of results illustrates the importance of choosing appropriate weights.

As one can see from Figure 2, we were unable to find a set of weights that allows the LP results to perfectly match those of ORM. For some basins, such as Lake Okeechobee and the water conservation areas, the ORM levels deviate significantly from the target levels. In general, these are the basins for which the LP results differ more widely from the ORM results. There is little reason to assume *a priori* that a set of weights can be found such that the LP levels deviate from the targets in precisely the same manner as in the ORM. In order to get the stages to match more closely, however, additional structure can be added to the model. This is discussed in more detail below.

The results for flows through the structures are shown in Figure 3.

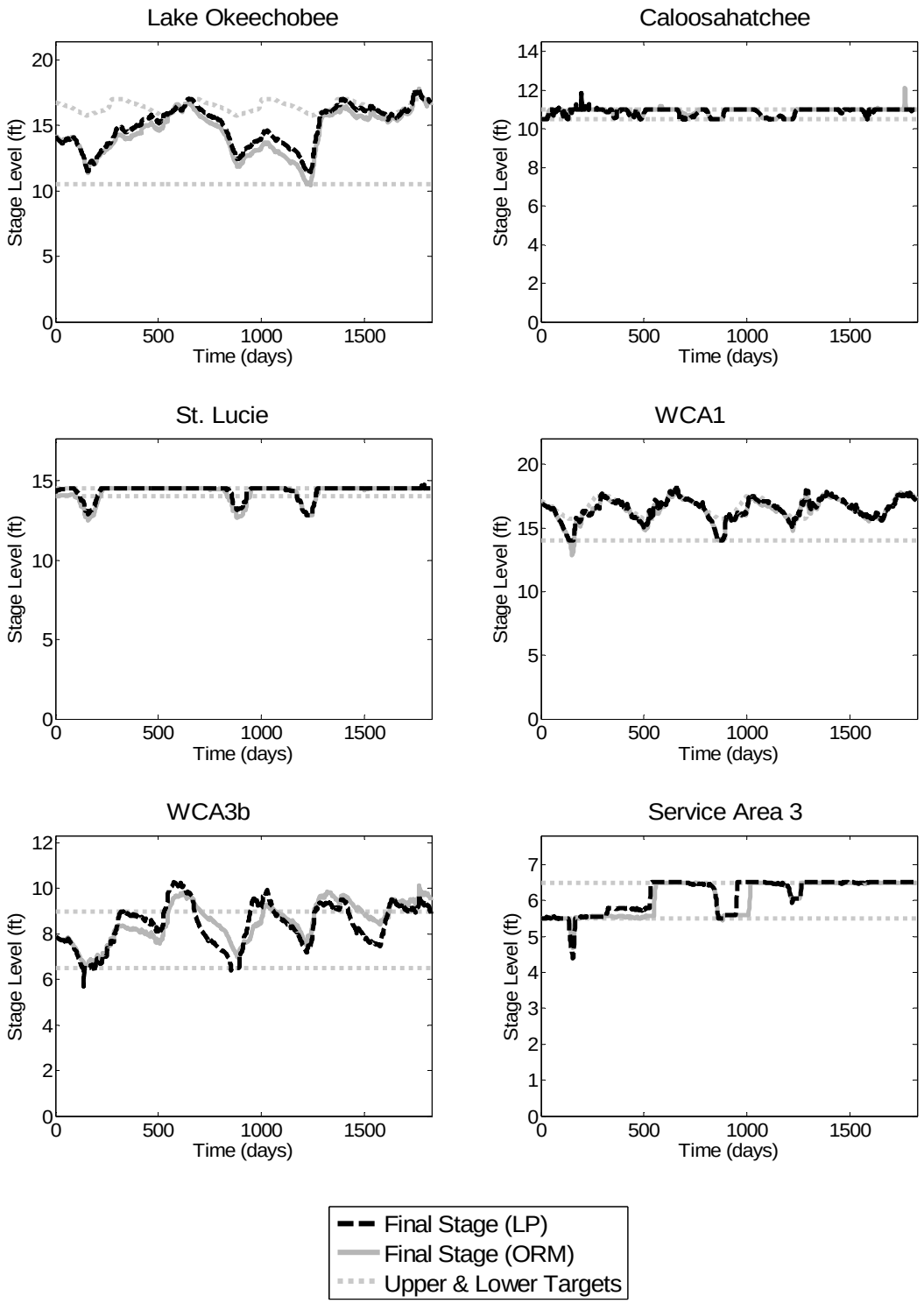


Figure 2 Stage results for LP model and ORM. The LP results are from the full version of the LP model, containing flow capacity constraints, management constraints, and minimum flow requirements. Selected basins only; full results shown in Appendix A.

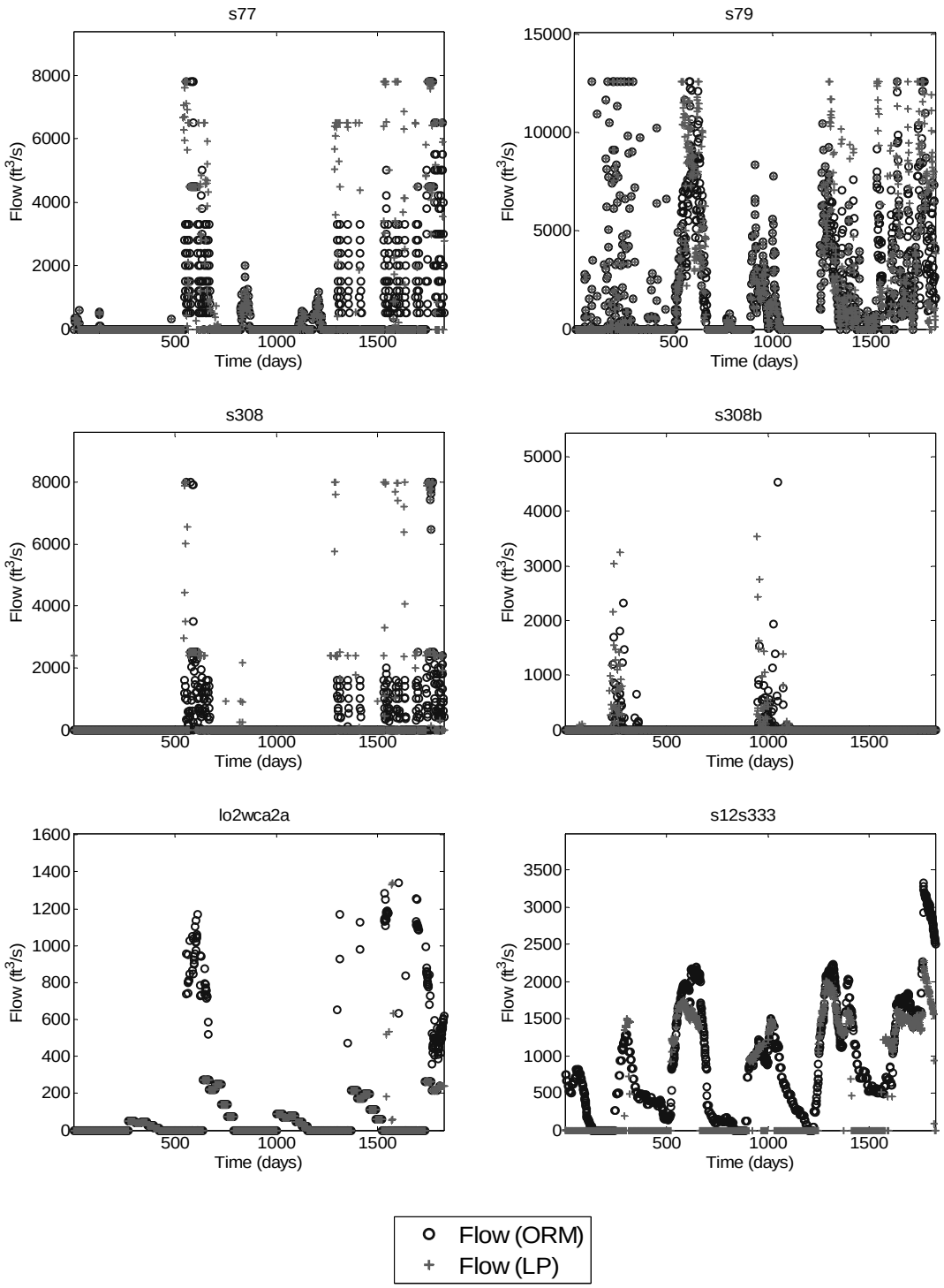


Figure 3 Flow results for LP model and ORM. The LP results are from the full version of the LP model, containing flow capacity constraints, management constraints, and minimum flow requirements. Selected structures only; full results shown in Appendix A.

A general characteristic of all of the results shown in this report is that *stage* results from both models seem to match each other more closely than do the *flow* results. This is not surprising, given the setup of the LP model. The objective function seeks to minimize deviations from target *stage* levels, and the weights on the deviations were calibrated in order to get the stage levels to match. While the flow capacity and management constraints do put some limits on the flow levels, there are not any flow targets in the model. Since the network is complex, with basins interconnected in multiple ways, there are a number of flow combinations that can achieve a particular set of basin stage levels. There is no guarantee that the LP model will choose the same flow routing methods as the ORM in order to achieve the same stage results.

3.1.2. Version 2: Excludes all flow constraints

The previous figures showed the results of the LP model that contained flow capacity constraints, minimum flow requirements, and management constraints. In order to determine the impacts of these flow constraints, we ran the model with all of them excluded. Figure 4 shows the results of this basic model that contains only targets and weights (penalties) to control the behavior of the model.

The results in Figure 4 show that removing the flow constraints from the LP model allows the basins to stay closer to the target stage levels than in the previous version. The LP stages also seem to match the targets more closely than do the ORM results. As a consequence, the LP and ORM results do not match each other as well as in the full LP model (Figure 2).

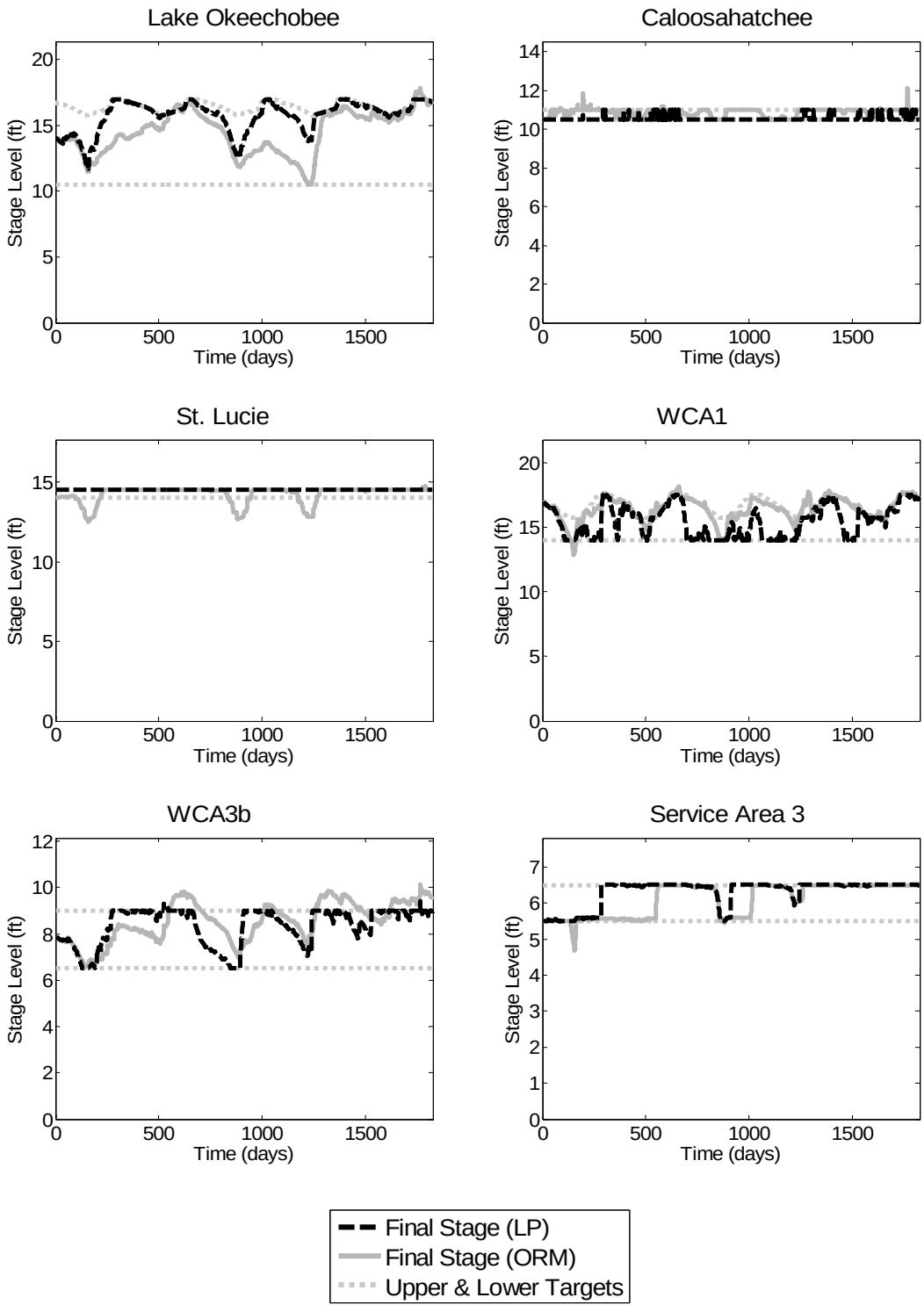


Figure 4 Stage results for LP model containing no constraints on structure flows; that is, there are no flow capacity constraints, no management constraints, and no minimum flow constraints. ORM results are shown for comparison. Selected basins only; full results shown in Appendix A.

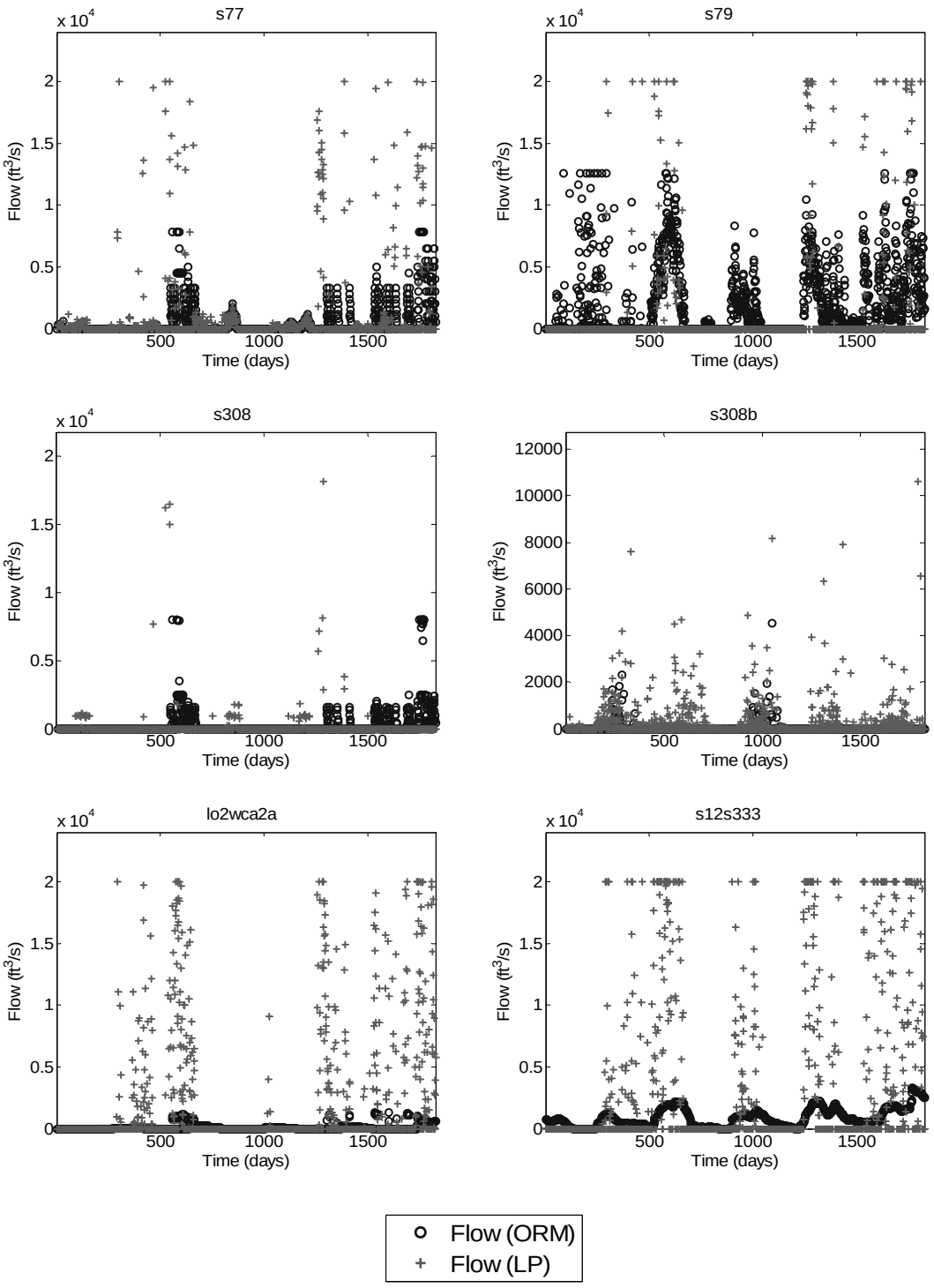


Figure 5 Flow results for LP model containing no constraints on structure flows; that is, there are no flow capacity constraints, no management constraints, and no minimum flow constraints. ORM results are shown for comparison. Selected structures only; full results shown in Appendix A.

Figure 5 shows the results of the structure flows for this version of the LP without flow constraints. A comparison between Figure 5 and Figure 3 gives insight into the impacts of the flow constraints.

One of the first things we notice from Figure 5 is that for most structures, the scale on the y-axis is significantly different from Figure 3. For this version, a default flow capacity of 2×10^4 ft^3/s was imposed for all structures. This is a much higher limit than the flow capacity constraints imposed in the previous LP version. As you can see from Figure 5, in the absence of these capacity constraints, many of the structures have flows approaching 2×10^4 ft^3/s. Since the flow capacity constraints generally represent the maximum amount of water that can physically flow through the structure, a model that allows flows in excess of this amount would be significantly flawed. Since the flow capacity constraints are relatively straightforward to implement in the LP model, and since they appear to be binding in many cases, it makes sense to implement them in the LP model.

3.1.3. Version 3: Includes flow capacity constraints, but excludes management constraints and minimum flow requirements

The next version of the LP model that we discuss enforces the flow capacity constraints, but continues to omit the management constraints and minimum flow requirements. We can compare these results to those of the full version of the LP in order to determine the effects of the management constraints and minimum flow requirements. The stage results are shown in Figure 6.

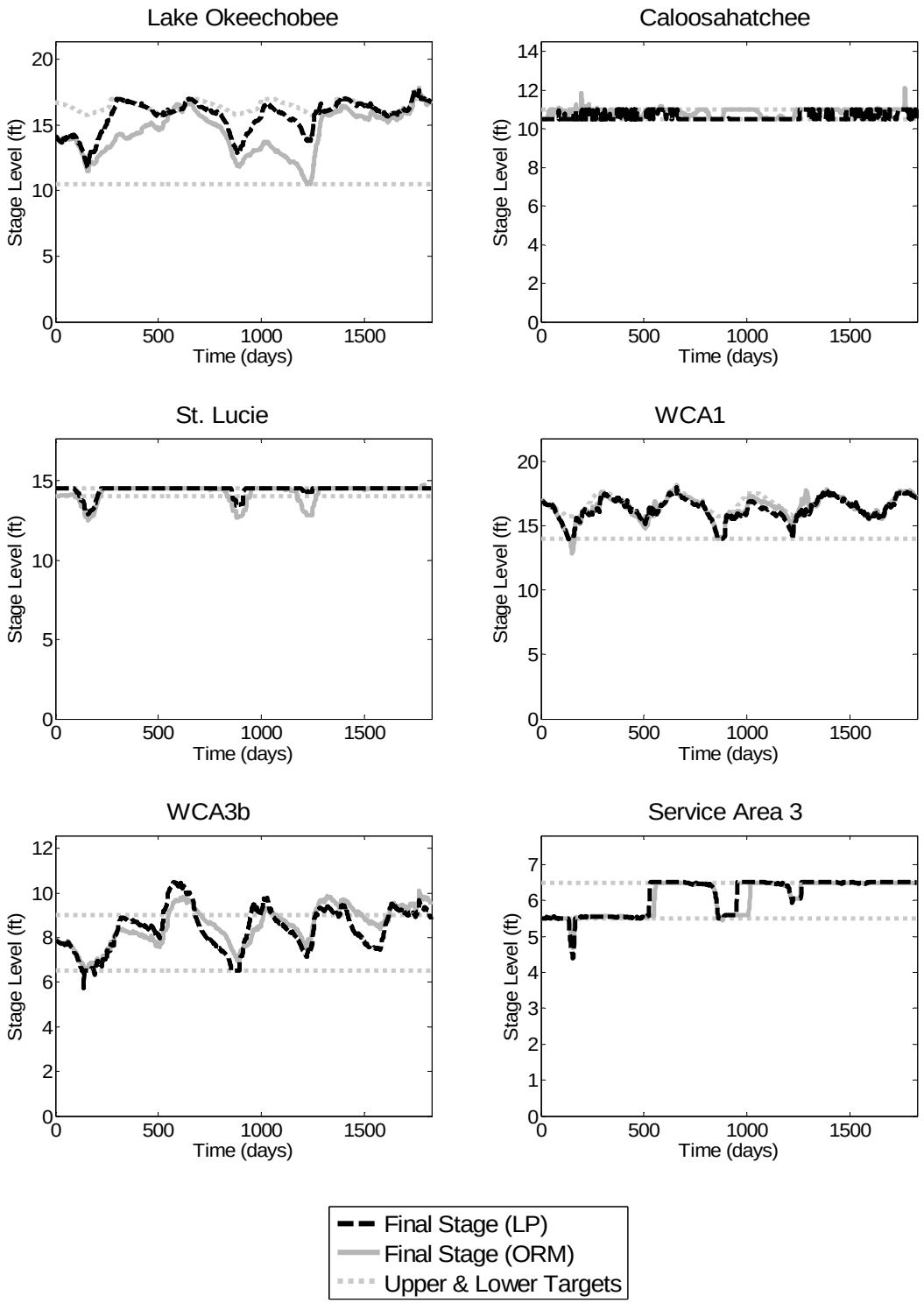


Figure 6 Stage results for LP model containing flow capacity constraints. Management constraints and minimum flow constraints are excluded. ORM results shown for comparison. Selected basins only; full results shown in Appendix A.

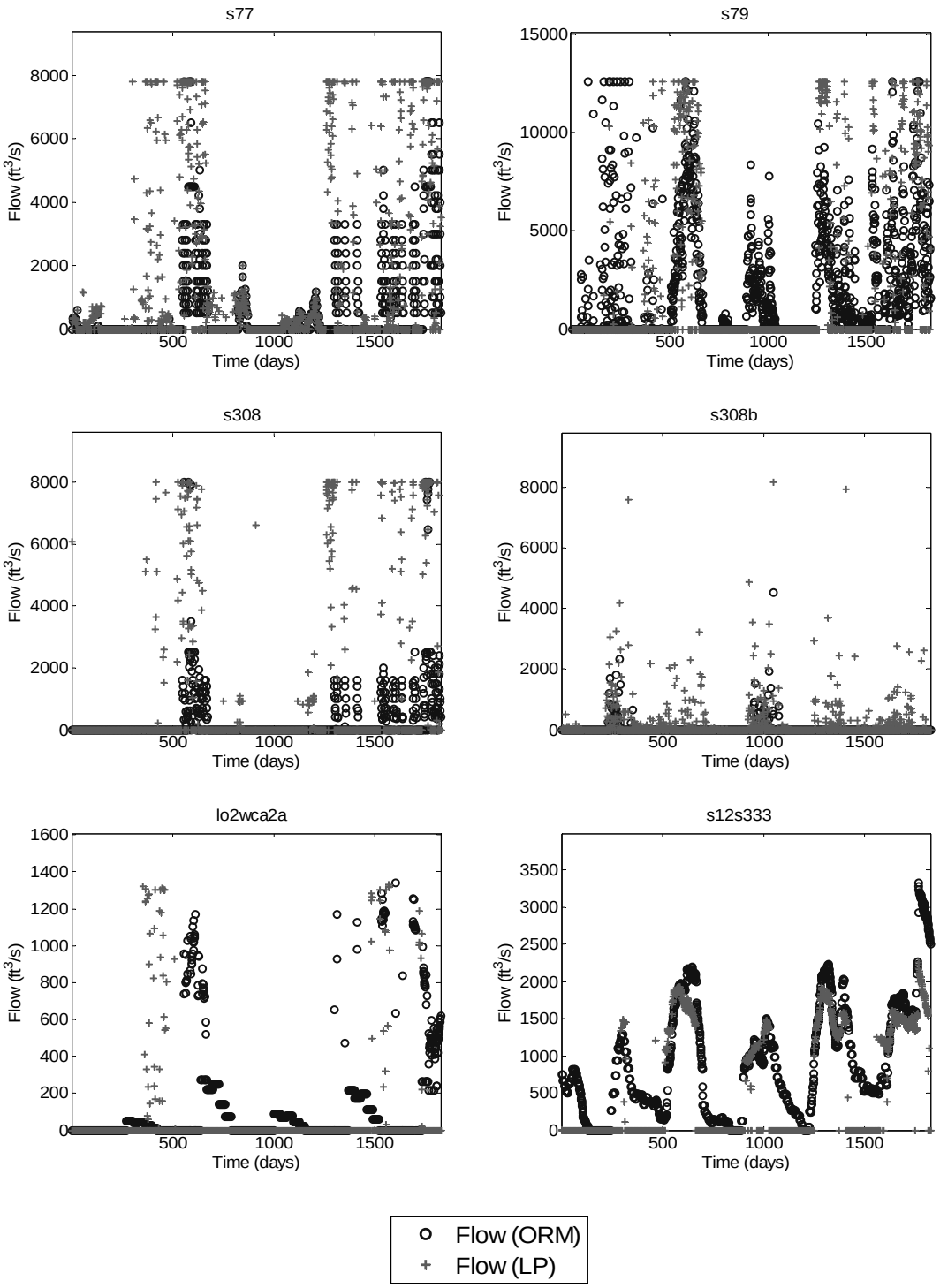


Figure 7 Flow results for LP model containing flow capacity constraints. Management constraints and minimum flow constraints are excluded. ORM results shown for comparison. Selected structures only; full results shown in Appendix A.

From the stage results in Figure 6, we notice that the LP and ORM levels generally match each other more closely than in the previous version (Figure 4), which excluded the flow capacity constraints. In particular, the largest differences between the two versions can be seen in the results for St. Lucie, the Water Conservation Areas, and Service Area 3. These differences indicate that the flow capacity constraints play a role in controlling the behavior of the model, and they guide the LP results towards those of the ORM. The same conclusion can be drawn from Figure 7, which shows the structure flow results.

It is also worth noting that the stage results for Lake Okeechobee do not seem to differ much between Figure 4 and Figure 6. However, there is a large difference between the results that contain management constraints and minimum flow requirements (Figure 2) and those that omit these constraints (Figure 6). The implication is that the management constraints and minimum flow requirements play a substantial role in guiding the LP results closer to those of the ORM.

There are also substantial differences in the flow results between the LP model versions that include and exclude management constraints; in particular, the results differ for most of the flow structures for which management constraints or minimum flow constraints are enforced.[20] The largest differences can be seen in the flows through the backpumping structures: S77bp, S308bp, and hgs5bp. There is a much greater occurrence of backpumping in the LP model without management constraints (Figure 7). The flow also varies significantly for structures S77 and S308, which manage the releases from Lake Okeechobee into the Caloosahatchee and St. Lucie

[20] Structures with management constraints: S77, S308, S77bp, S308bp, hgs5bp, S10, and S343. Structures with minimum flow requirements: lo2wca1, lo2wca2a, lo2wca3a.

Rivers, respectively. The impacts of the minimum flow requirements can be seen in the results for structures lo2wca1, lo2wca2a, and lo2wca3a. The minimum flow levels are relatively small (under 500 ft^3/s) compared to the highest flows through these structures. In the model containing minimum flow requirements (Figure 3), the LP and ORM flows correspond for low levels of flow where the minimum flow constraints are binding. This is not the case in the model excluding minimum flow requirements (Figure 7).

3.2. Discussion and Future Work

From this investigation we can conclude that the LP model does a reasonable job of matching the stage levels simulated by the ORM. There are noticeable discrepancies between the stage levels for those basins that tend to deviate significantly from their targets: namely, Lake Okeechobee and the Water Conservation Areas. In addition, the simulated structure flows do not match as closely as the stages – for reasons previously discussed.

Our comparison of the different versions of the LP model indicates that the additional physical and operational constraints help to guide the model towards a more realistic outcome. With the flow capacity constraints imposed, the flow results from the LP model remain within their physical limits, and the LP flows correspond more closely to the ORM flows. Similarly, the management constraints ensure that the flows through particular structures obey particular rules, such as limits on backpumping or the releases from Lake Okeechobee.

The success of the flow constraints in controlling the structure flows shows how explicit rules can be used to constrain and add structure to the optimization model.

While the constraints that have already been implemented cause the LP flow results to match those of the ORM more closely, there remain important discrepancies between the two models for both the stage and flow results.

In order to force the LP model to more closely match the ORM, a number of steps can be taken. One step is to change the target levels in the LP model. The targets chosen for the LP model described here were taken from the ORM; as mentioned earlier, the difference in structure between the LP model and the ORM means that the same targets may not be appropriate for both models.

Another approach would be to add additional targets. Each basin can have any number of target levels. In the present LP model, there are two target levels. Once the stage is below the upper target, further decreases in the stage increase the total penalty for that basin, and this total penalty increases at a constant rate. As soon as the stage falls below the second target, there is an increase in the rate at which the total penalty increases. In this manner, multiple targets can approximate a nonlinear penalty function, where there is an increase in the rate at which the total penalty changes as the basin stage falls. This can add stability to the model by (for example) causing the effects of a water shortage to be spread across more basins, rather than having one or two basins drained fully.

Another way to accomplish the same goals is by penalizing the *maximum deviation* among all the basins. This will (generally) ensure that the deviations among all basins are as close to being equal as possible. Penalizing the *maximum weighted deviation* ensures that the ratio of deviations between two basins will be proportional to their relative weights. Alternatively, one can group the basins into subsets, and penalize the

maximum deviation within each group, then sum all of these penalties in the objective function. In addition, the penalties on maximum deviations can be included in the objective function along with penalties on deviations for individual basins. All of these possibilities provide ways to add more structure and control to the LP model.

Another way to adjust the results of the LP model would be to add additional management constraints. The management constraints implemented in the current version of the LP model represent only a subset of the constraints imposed by the coordinators in the ORM. Implementing more of these coordinators would likely bring the LP results closer in line with those of the ORM. Another option could be to introduce flow penalties for some of the structures, such that the flow through a particular structure is multiplied by a particular weight, and this value is added to the objective function. The relative size of the penalties can be used to discourage flow through particular structures and achieve desired flow results.[21] Alternatively, instead of penalizing any flow through the structure, one could introduce flow targets and only penalize deviations from those targets.

While we were able to implement some of the functionality of the ORM coordinators, we did not implement any of the ORM assessors. Integrating this information into the LP is likely to produce better results as well. In order to fully integrate information from the coordinators or assessors into the LP model, it will be necessary to address the different types of flow in the ORM. The LP currently makes no distinction between water supply flow and flood control flow, and so it is not possible to impose constraints on only one type. The ORM distinguishes between both types of flow, and

[21] This functionality is already implemented in the current LP model, but all of the flow penalties are set to zero so that they have no effect.

imposes constraints on each type separately. In order to make further progress towards implementing aspects of the coordinators or assessors in the LP model, one would need more information than was available to us concerning the ways in which the ORM treats each type of flow and how it combines these types to produce the final output.

It may be possible to tackle this problem in a number of ways. One way would be to examine the rules that are imposed on water supply and flood control flows in the ORM, and rewrite these rules such that they are in an appropriate form to impose on total flow in the LP. Another way to tackle this problem might be to create two separate LP models: one for flood control flow and one for water supply flow. It would then be up to the ORM assessors to determine the final flow. Yet another option would be to create a single LP model that includes both types of flow. It is not yet clear which of these options would be most feasible or useful. Addressing this question was beyond the scope of the work being described here, but represents an important next step.

When deciding how best to structure the LP and integrate it within the RSM, it is useful to consider what happens in the real world. The purpose of all of these programs is to model the actual responses of the South Florida region. Given a certain set of hydrological inputs (whether historically accurate or hypothetical), these models should simulate the outcomes that would actually occur. So in terms of management responses, it is important to consider how the water managers ("operators") are making their decisions. If there exists an explicit hierarchy of rules governing the flow decisions for all of the structures, then it is likely that the best way to model such a situation would be to explicitly code these rules into a simulation model. On the

other hand, it may be that the operators take a more holistic approach to managing the region, by observing the conditions in all of the water bodies and determining the flows in a ways that helps to best maintain the target levels. This sort of situation can be well-modeled by an optimization approach.

It may be that the actual situation in South Florida is a combination of the two situations that we have described. There may be explicit rules governing flows through some of the structures, but not all of them. In this case, the best simulation approach may be one that integrates both optimization and rule-based decision-making. For the basin-scale network described in this report, the set of rules may be explicitly known already, and the optimization may be unnecessary. However, if RSM is to be expanded to model the complex network of canals throughout South Florida, it may be too difficult to develop the hierarchy of rules for governing flows. This is the sort of situation where optimization can prove useful, eliminating the need to develop rules for each structure. The integrated approach would allow rules to be imposed only when necessary.

3.3. ***Conclusions***

1. This thesis has demonstrated how a Linear Programming (LP) optimization model can be used to simulate management decisions in South Florida. We have modeled the simplified network of the Object Routing Model (ORM) and compared the results from these two models.
2. This thesis has shown that including additional constraints on flows in the LP model causes the LP results to more closely match those of the ORM for both stages and flows. The importance of the weights in controlling the LP results has also been shown.

3. A number of ways have been suggested to force the LP results to more closely match those of the ORM: including additional stage targets; grouping basins and penalizing the *maximum* deviation of all the basins in the group; including additional flow constraints that already exist in RSM; or adding flow targets.
4. We have also briefly discussed the barriers to implementing more of the ORM functionality in the LP model, including the distinction made in the ORM between flow for water supply and flow for flood control and the lack of documentation about how flow quantities are decided. Further investigation is required in order to overcome these barriers.
5. We have shown how the linear optimization can be combined with rule-based constraints to control flows. This provides some insight into the ways in which LP methods can be integrated into RSM without limiting the functionality of RSM or discarding RSM components that have required significant investments to develop. An LP component can help to model management decisions by determining flow levels through some or all structures without requiring a dramatic restructuring of RSM or its Management Simulation Engine.

APPENDIX A

FULL RESULTS

This Appendix contains the full set of figures for the results of the three versions of the LP model described in the report. The three versions are: the full LP model containing the three types of flow constraints; a version containing none of the flow constraints; and a version containing the flow capacity constraints but omitting the management constraints and minimum flow constraints.

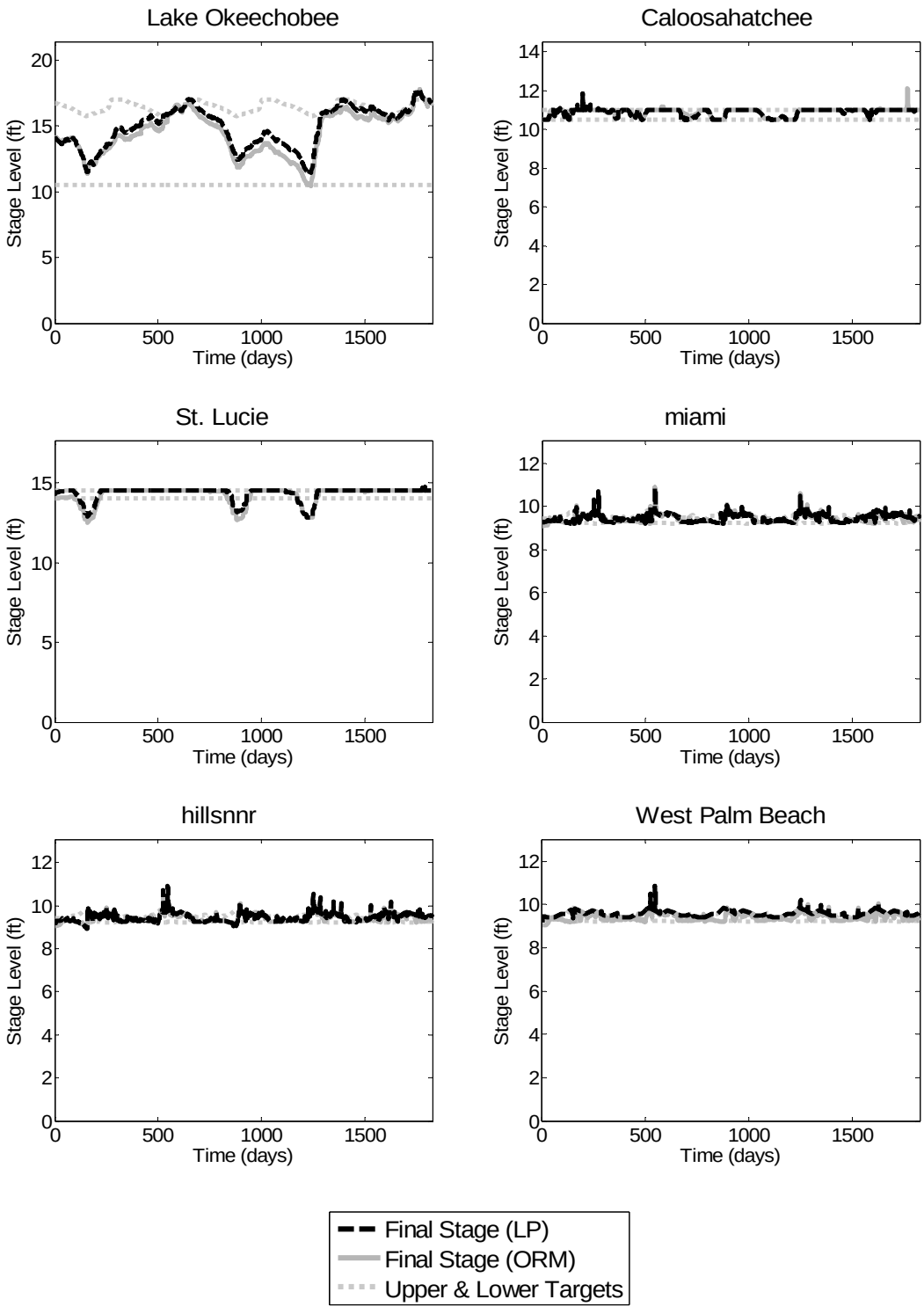


Figure A.1 Stage results for LP model and ORM. The LP results are from the full version of the LP model, containing flow capacity constraints, management constraints, and minimum flow requirements.

Figure A.1 (Continued)

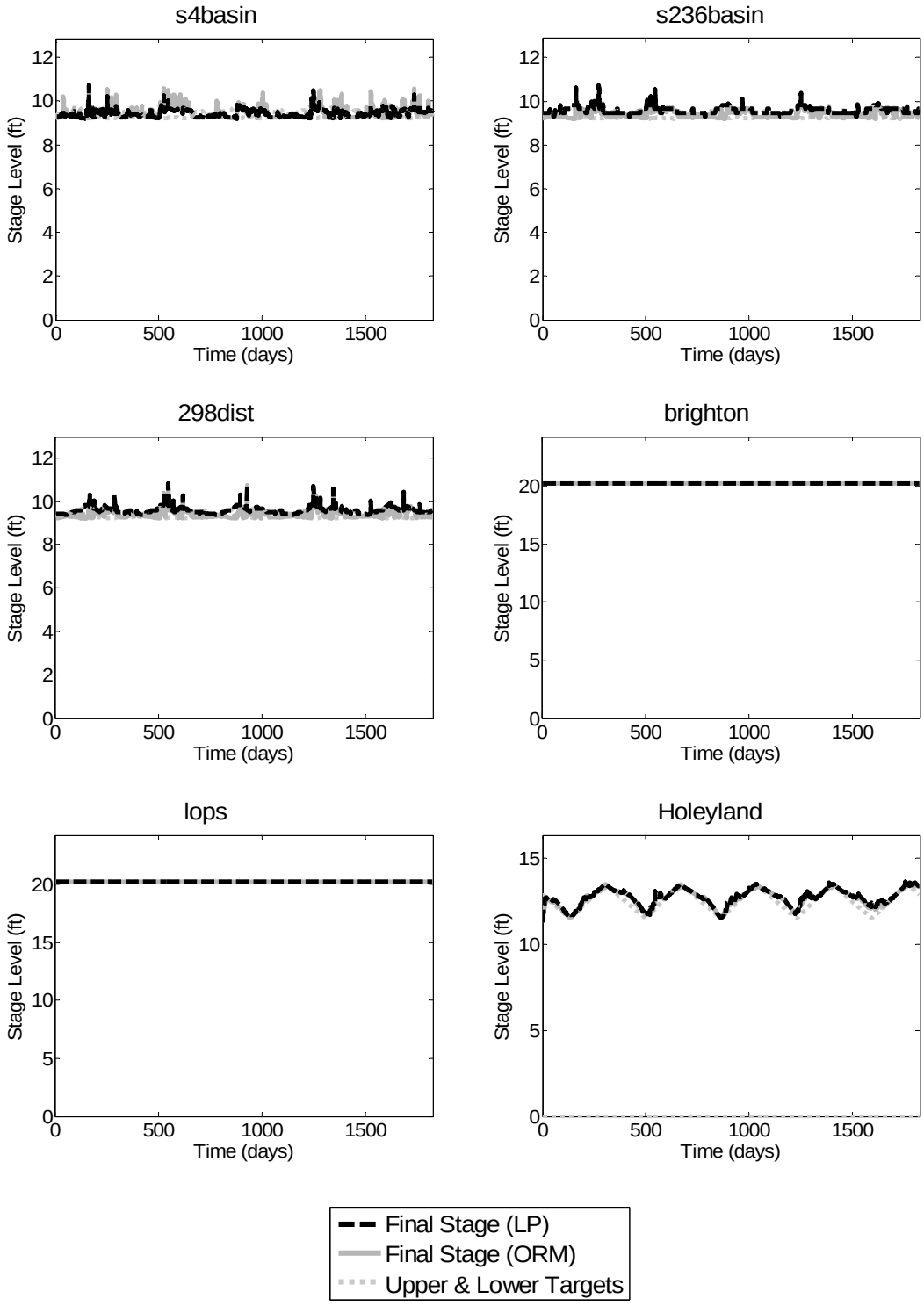


Figure A.1 (Continued)

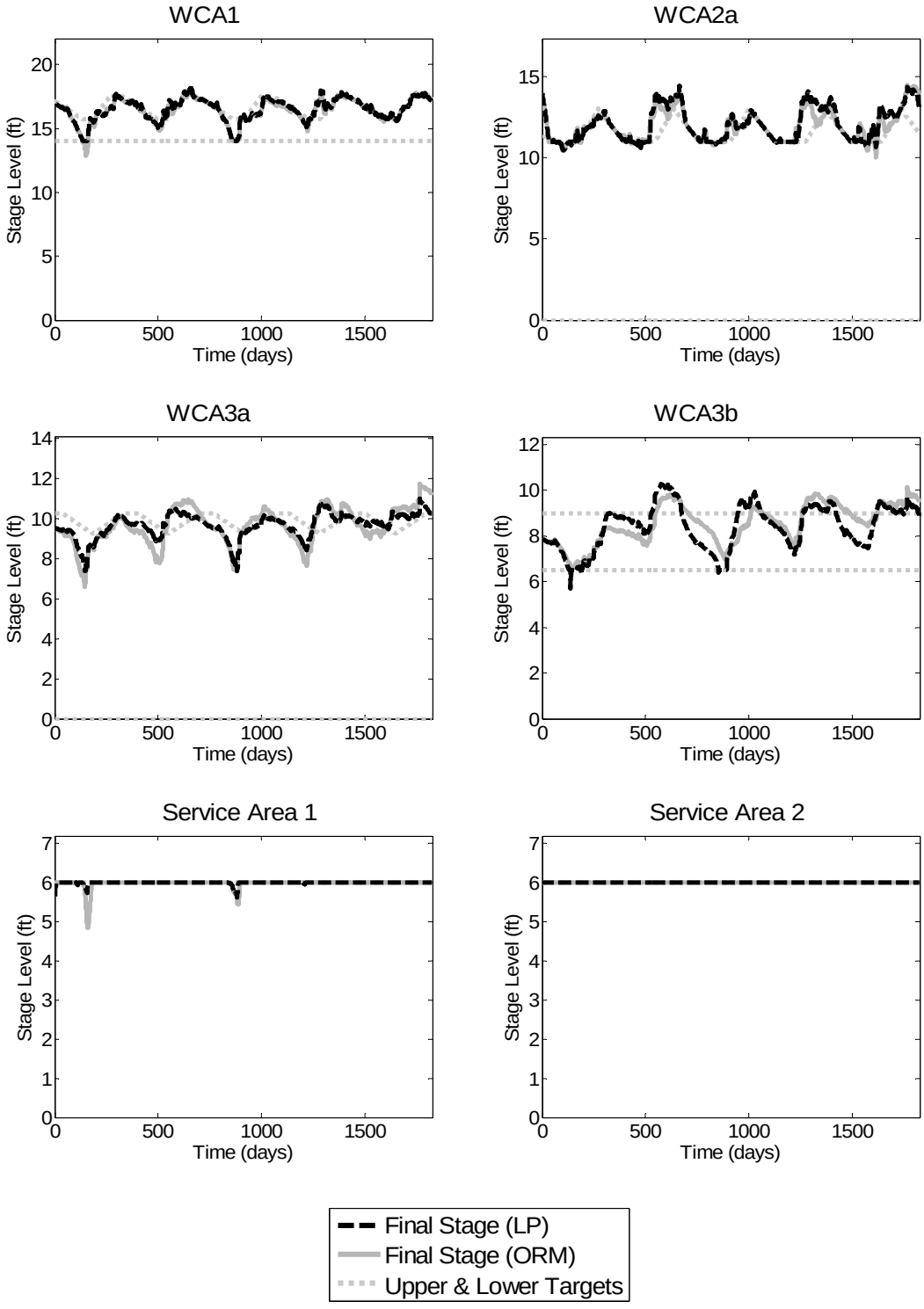
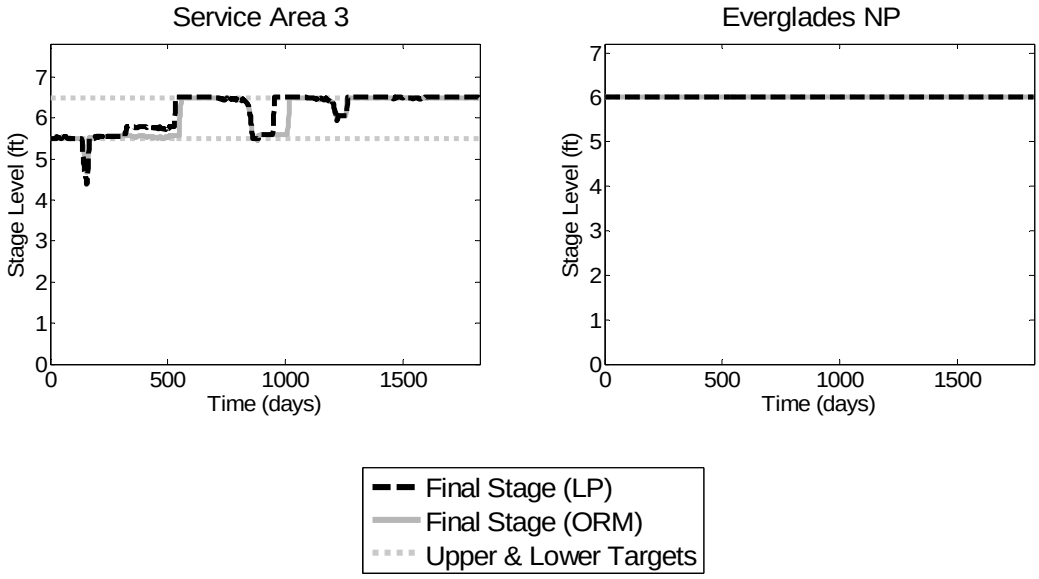


Figure A.1 (Continued)



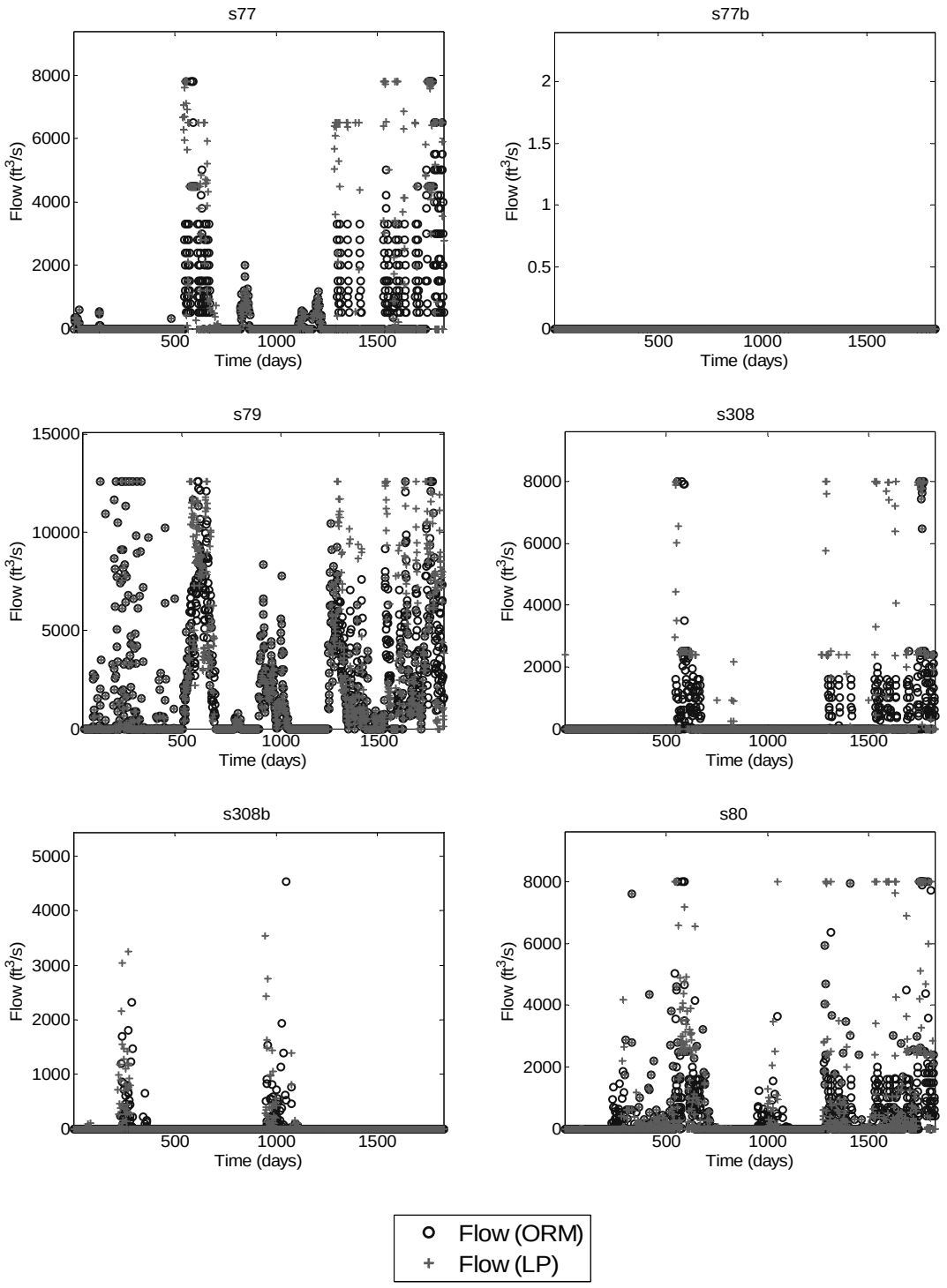


Figure A.2 Flow results for LP model and ORM. The LP results are from the full version of the LP model, containing flow capacity constraints, management constraints, and minimum flow requirements.

Figure A.2 (Continued)

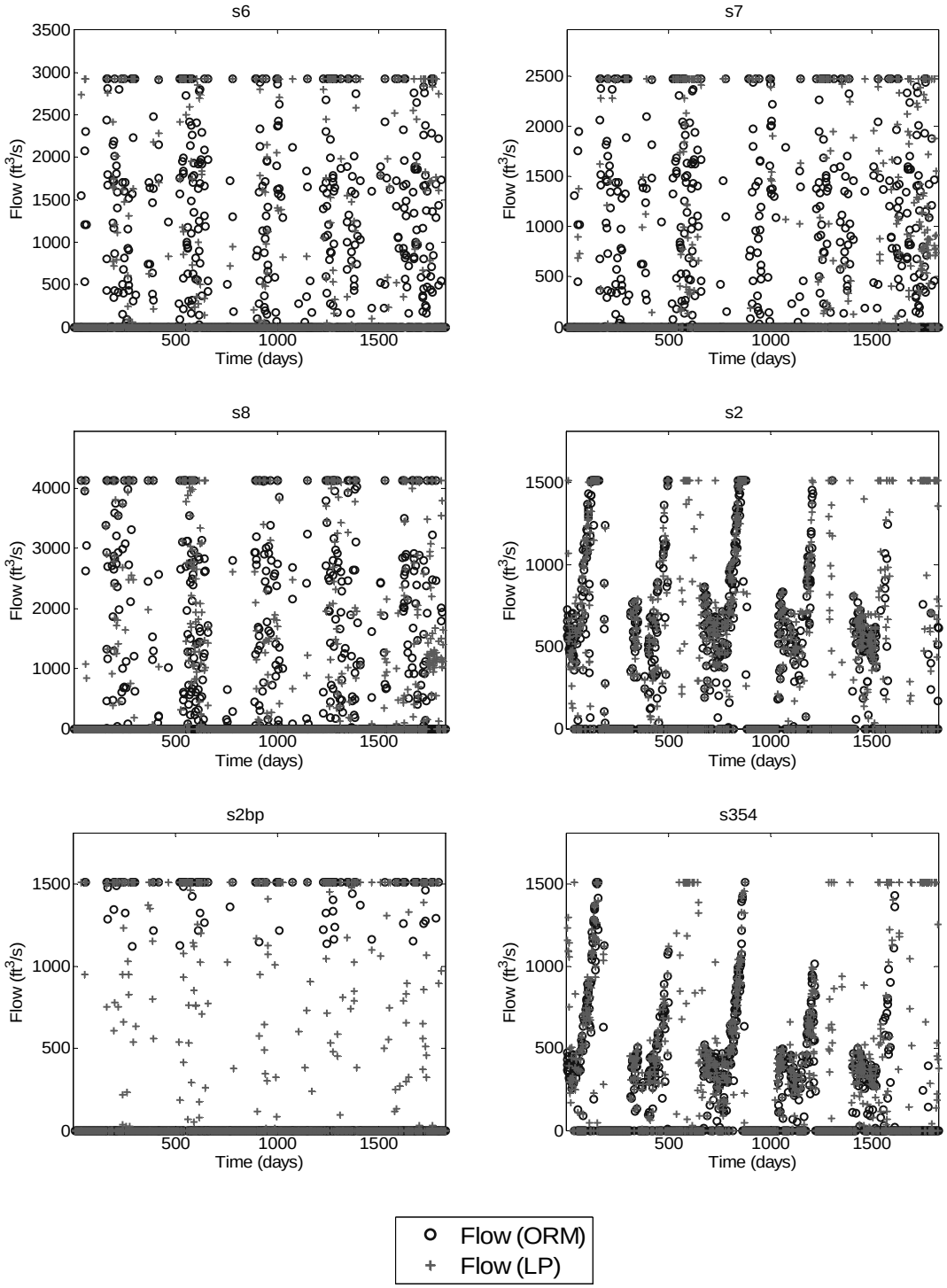


Figure A.2 (Continued)

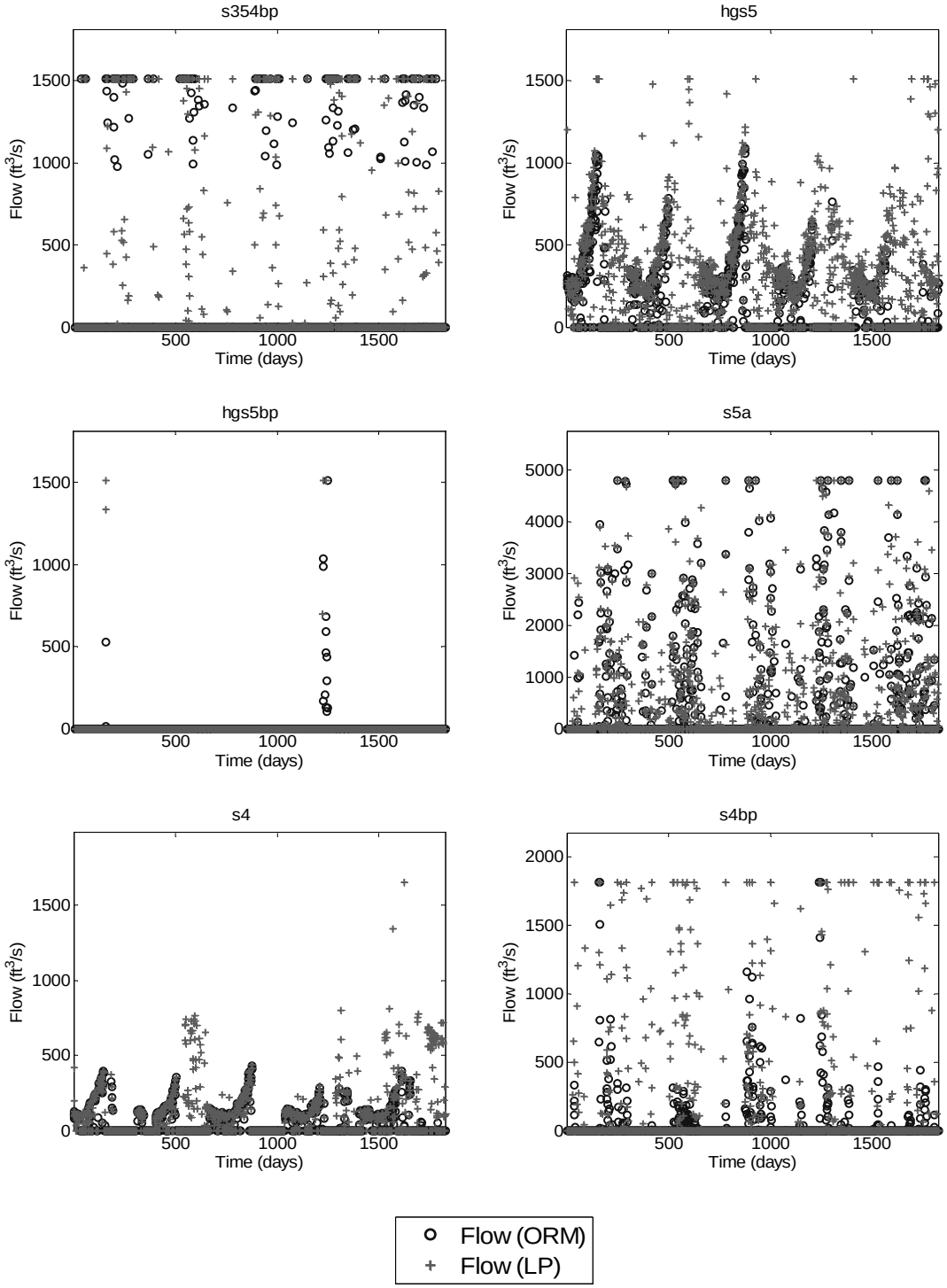


Figure A.2 (Continued)

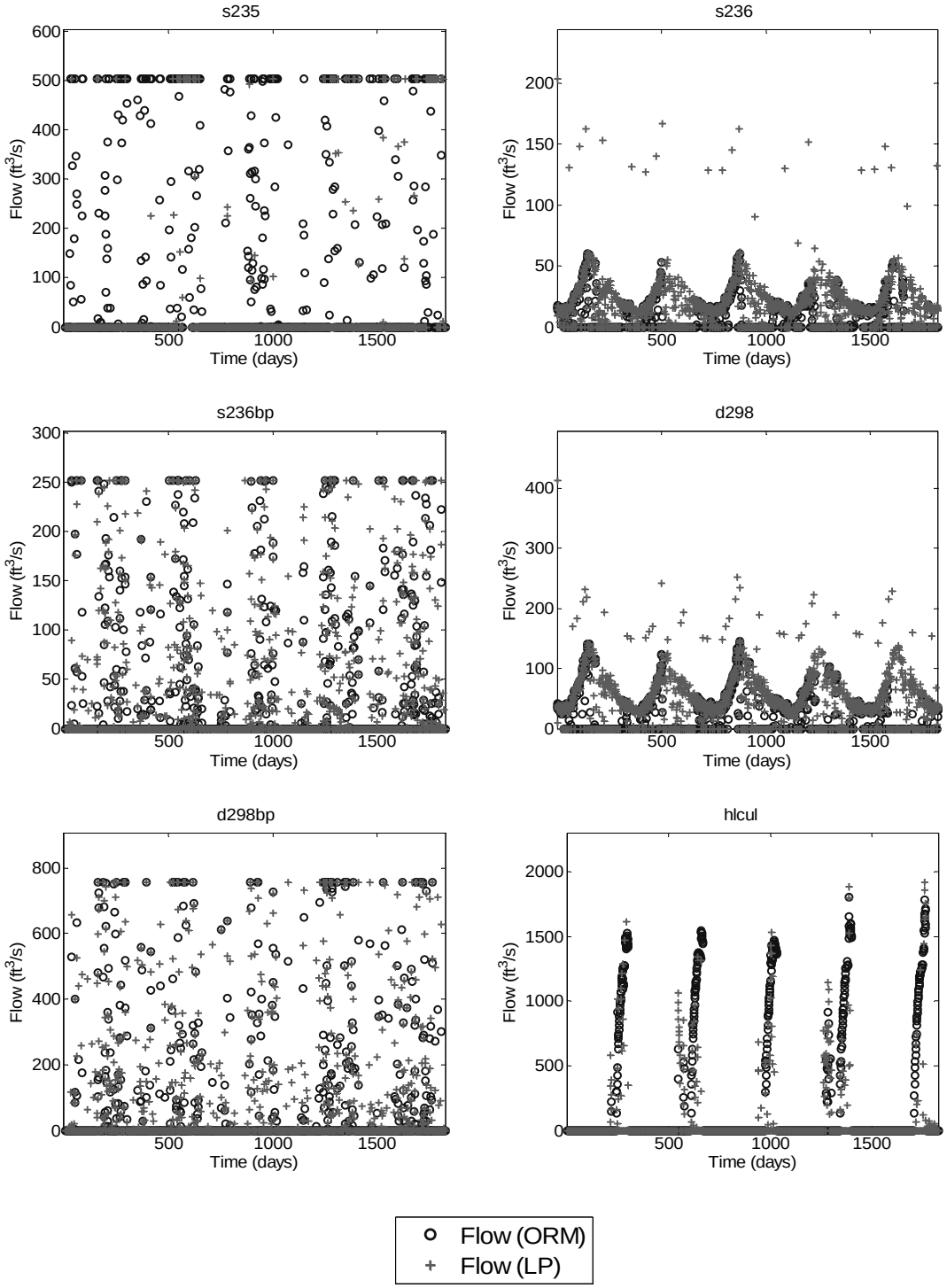


Figure A.2 (Continued)

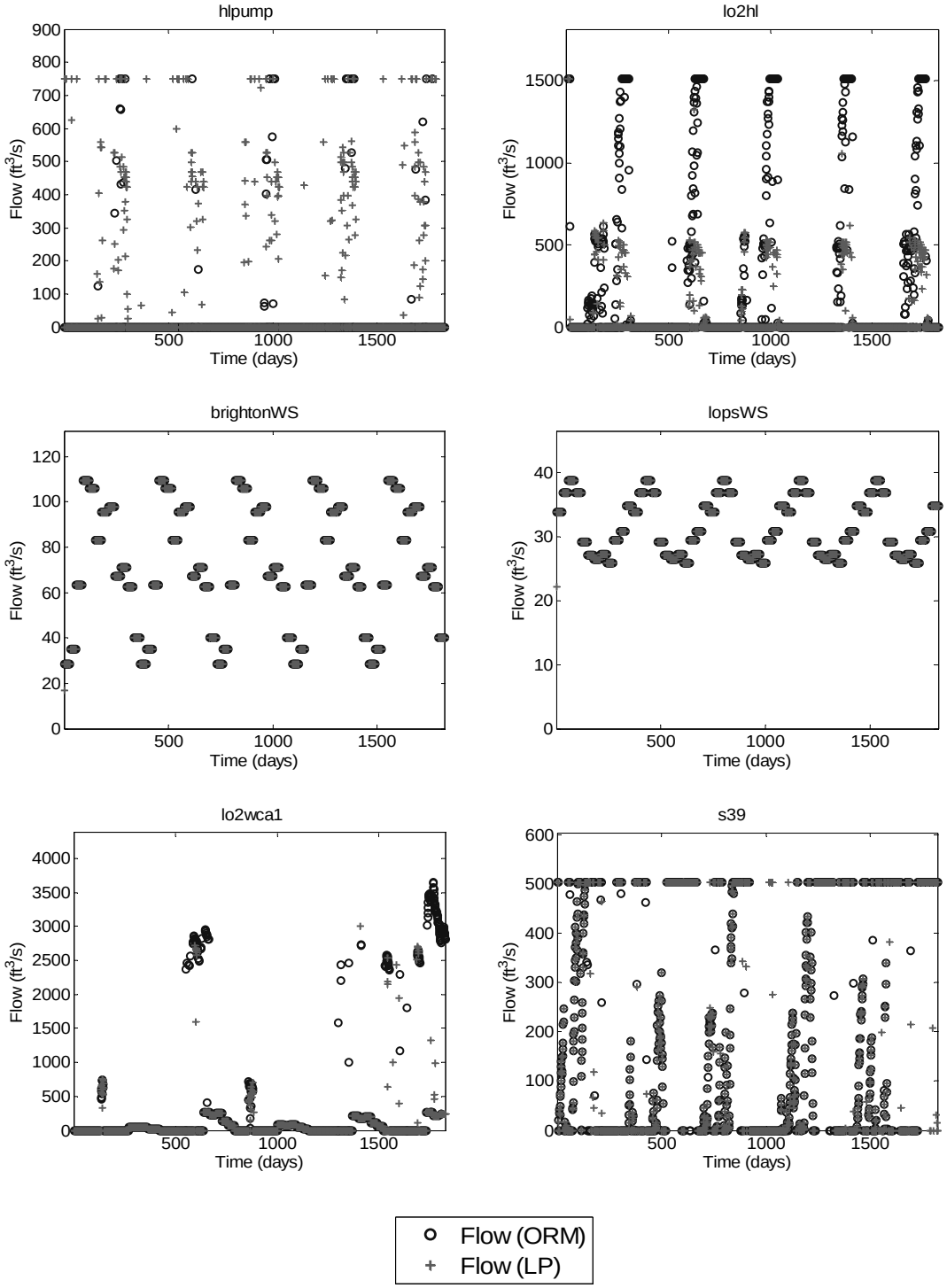


Figure A.2 (Continued)

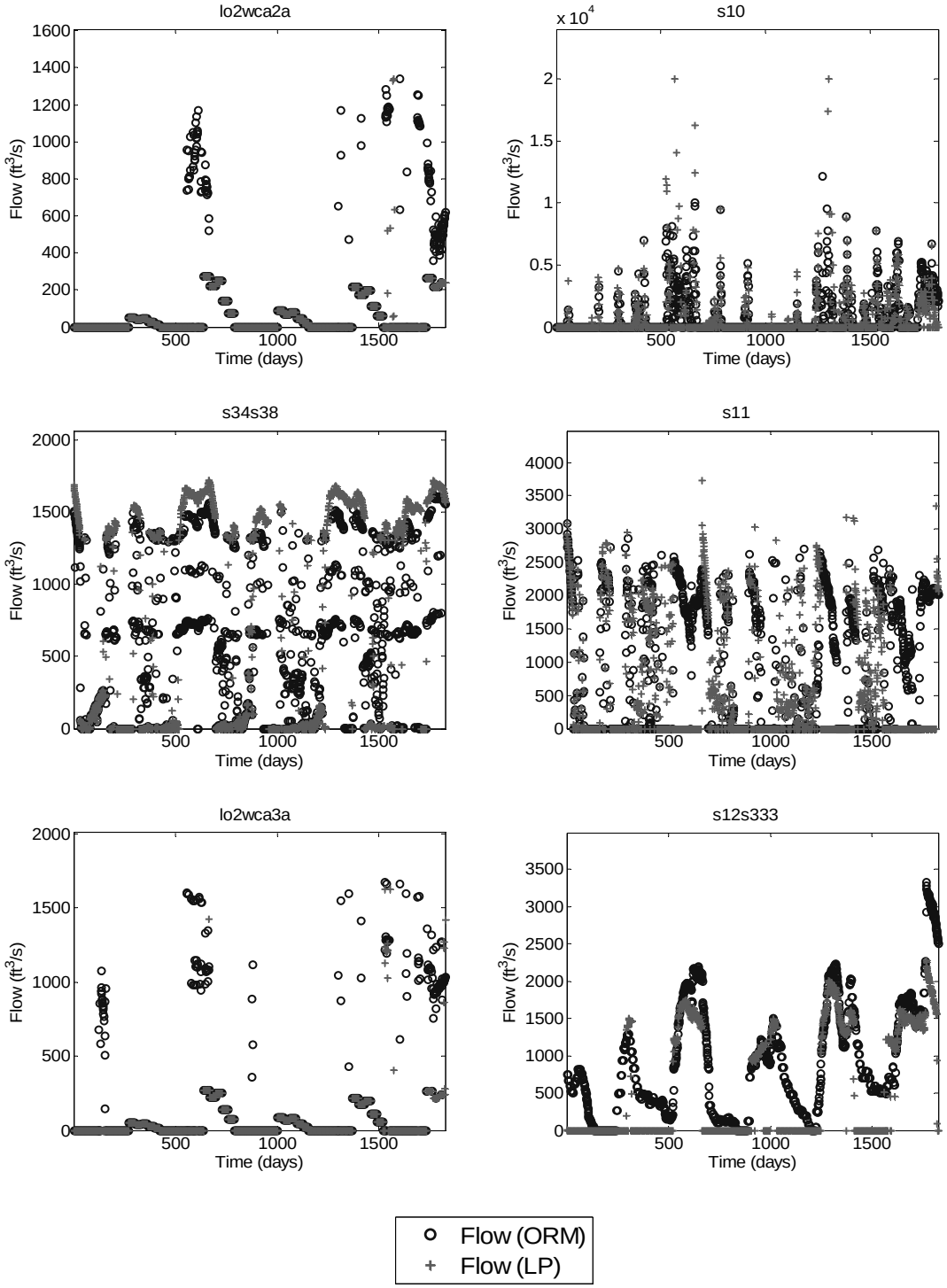
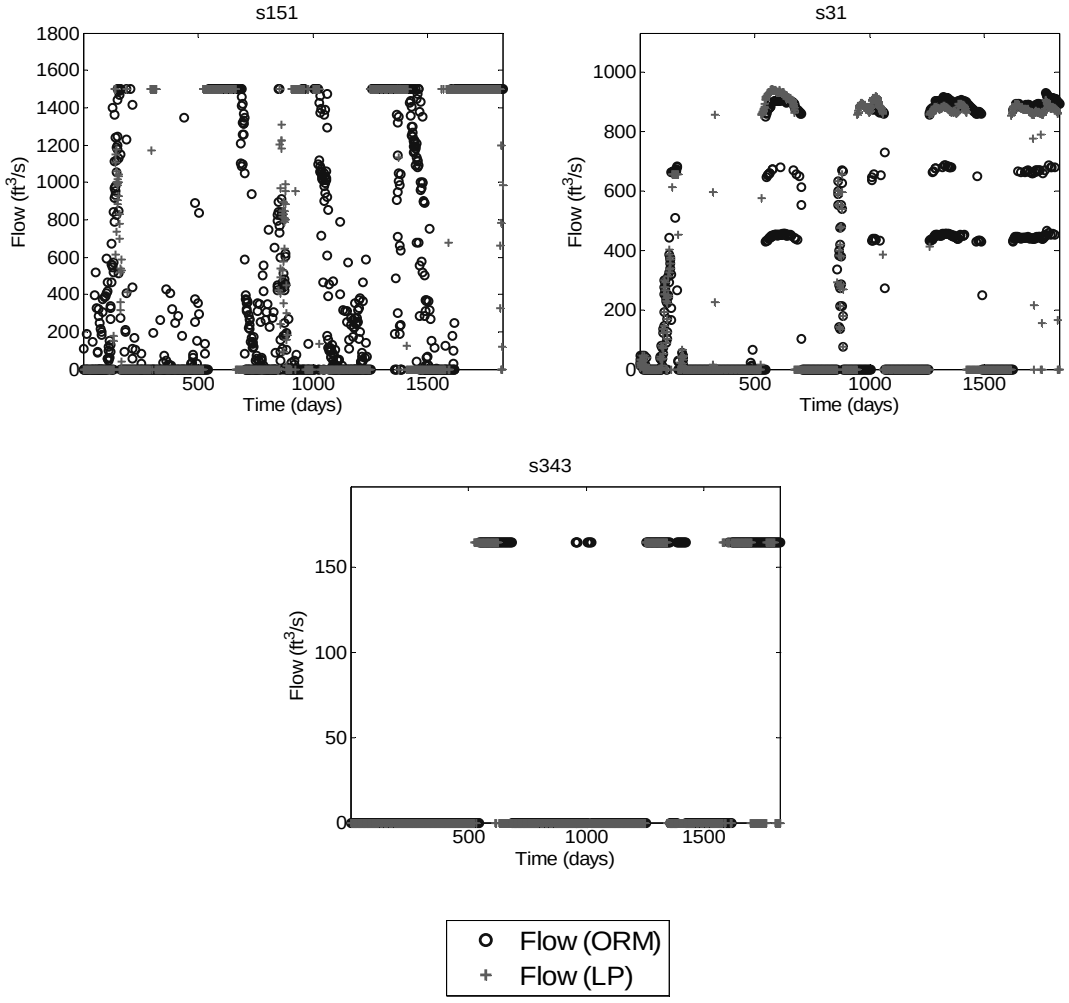


Figure A.2 (Continued)



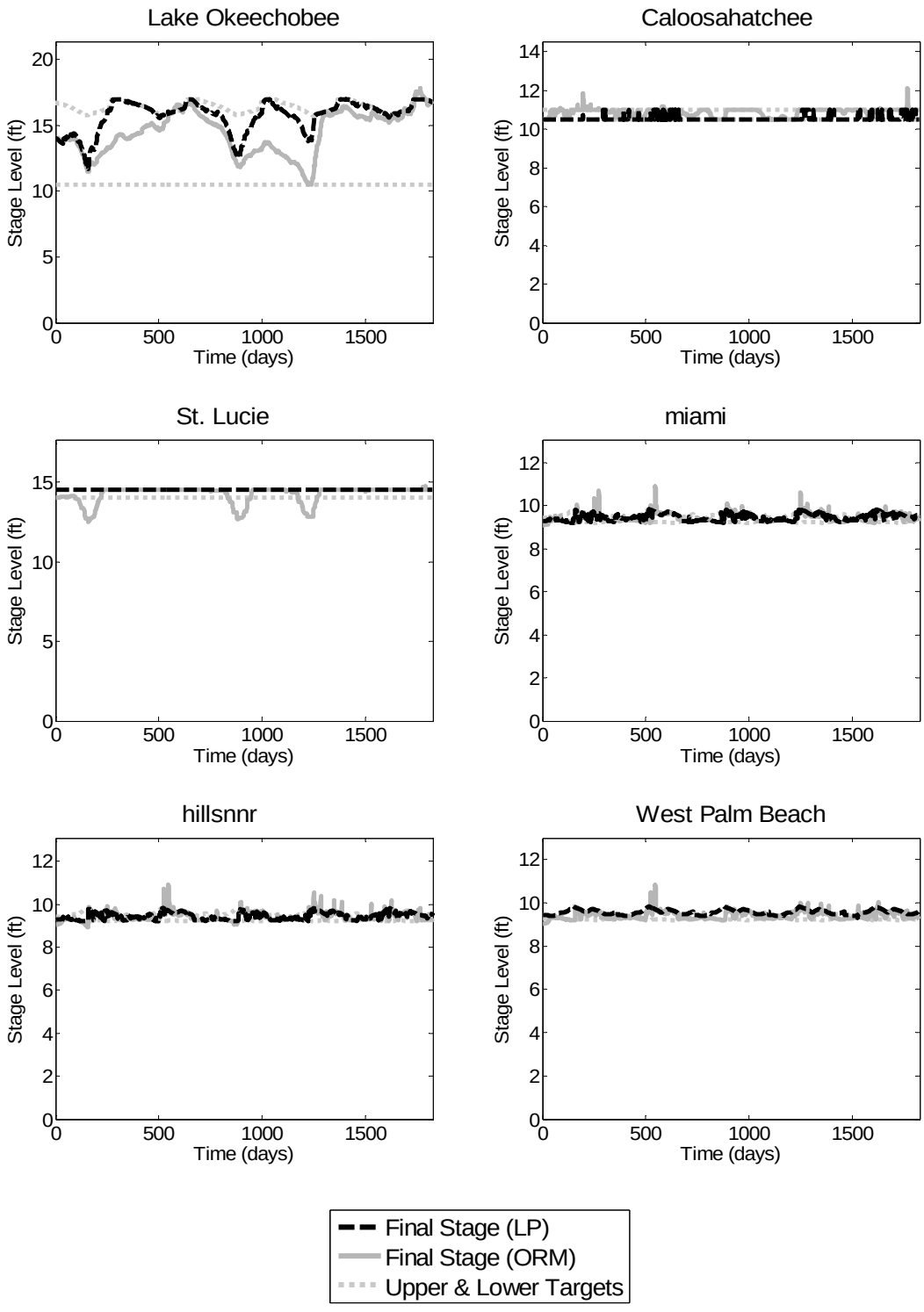


Figure A.3 Stage results for LP model containing no constraints on structure flows; that is, there are no flow capacity constraints, no management constraints, and no minimum flow constraints. ORM results are shown for comparison.

Figure A.3 (Continued)

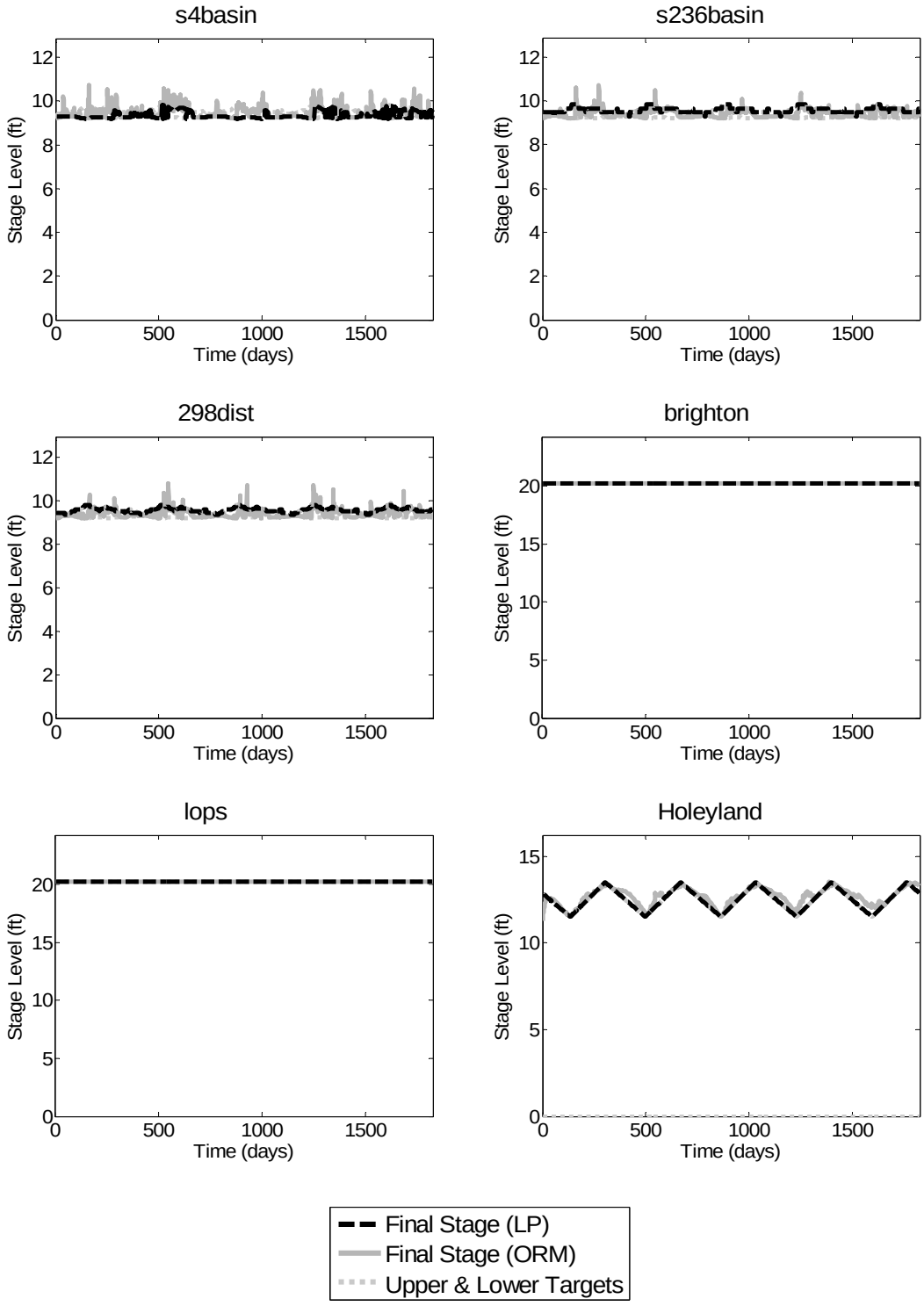


Figure A.3 (Continued)

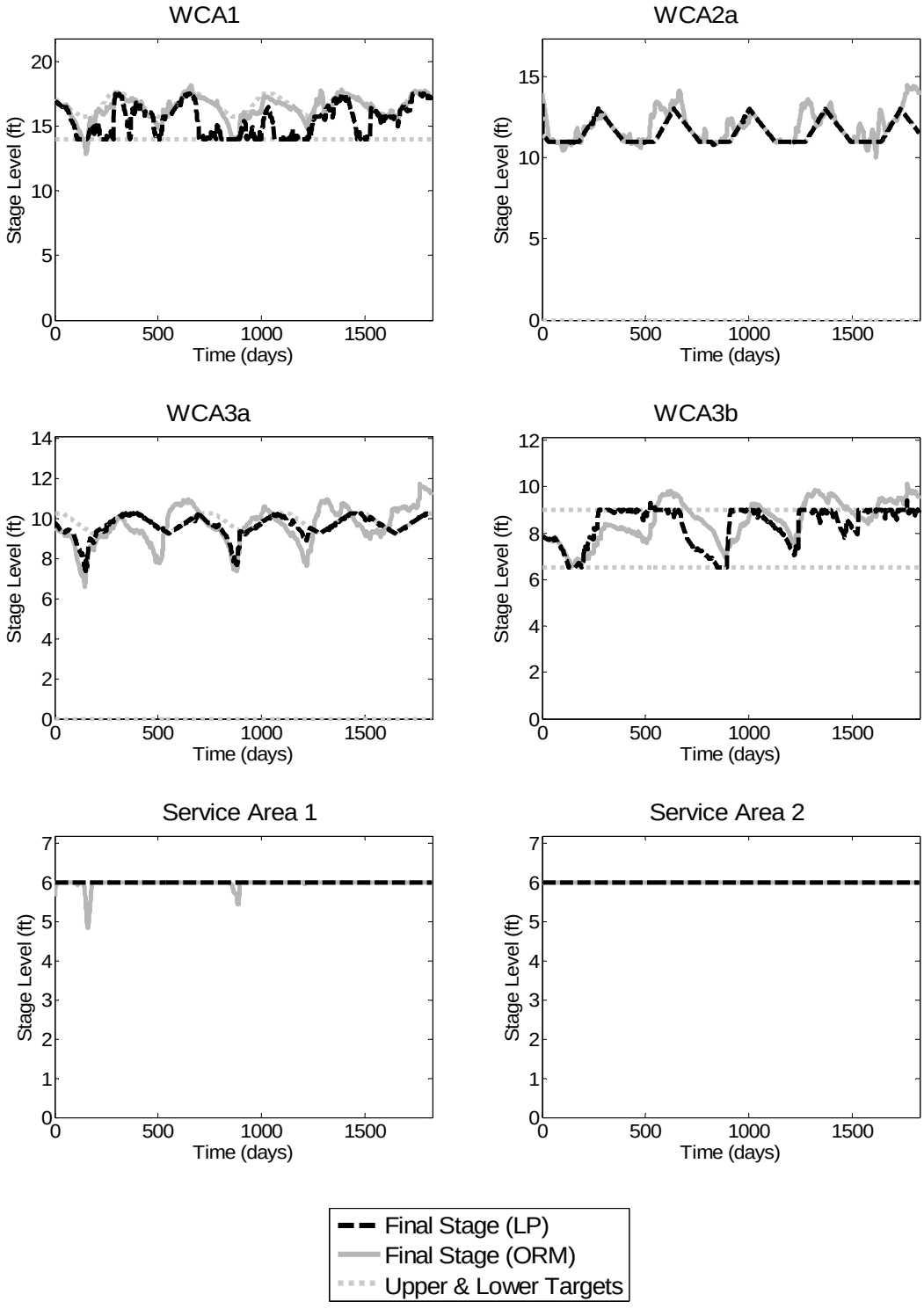
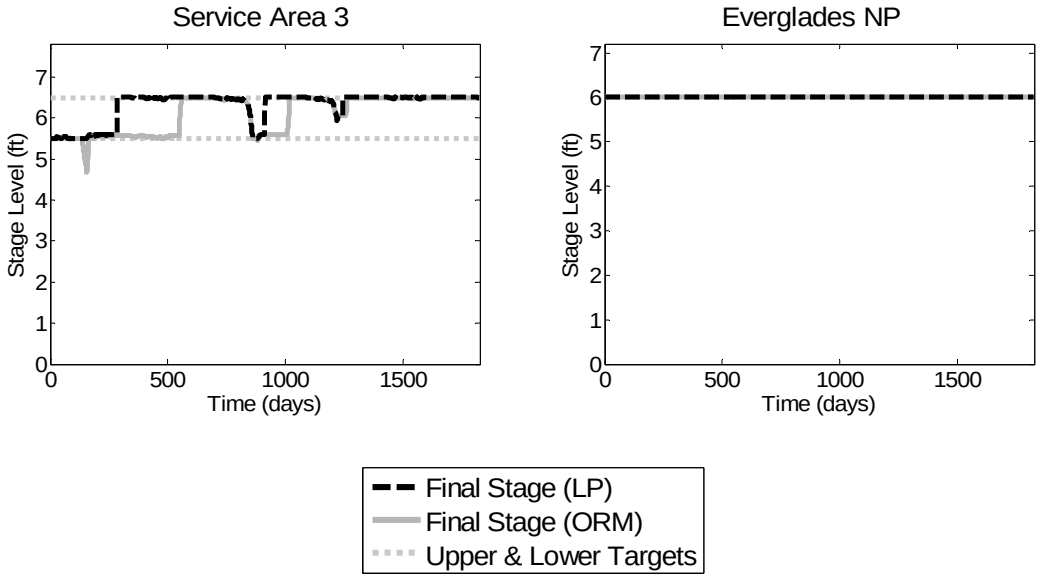


Figure A.3 (Continued)



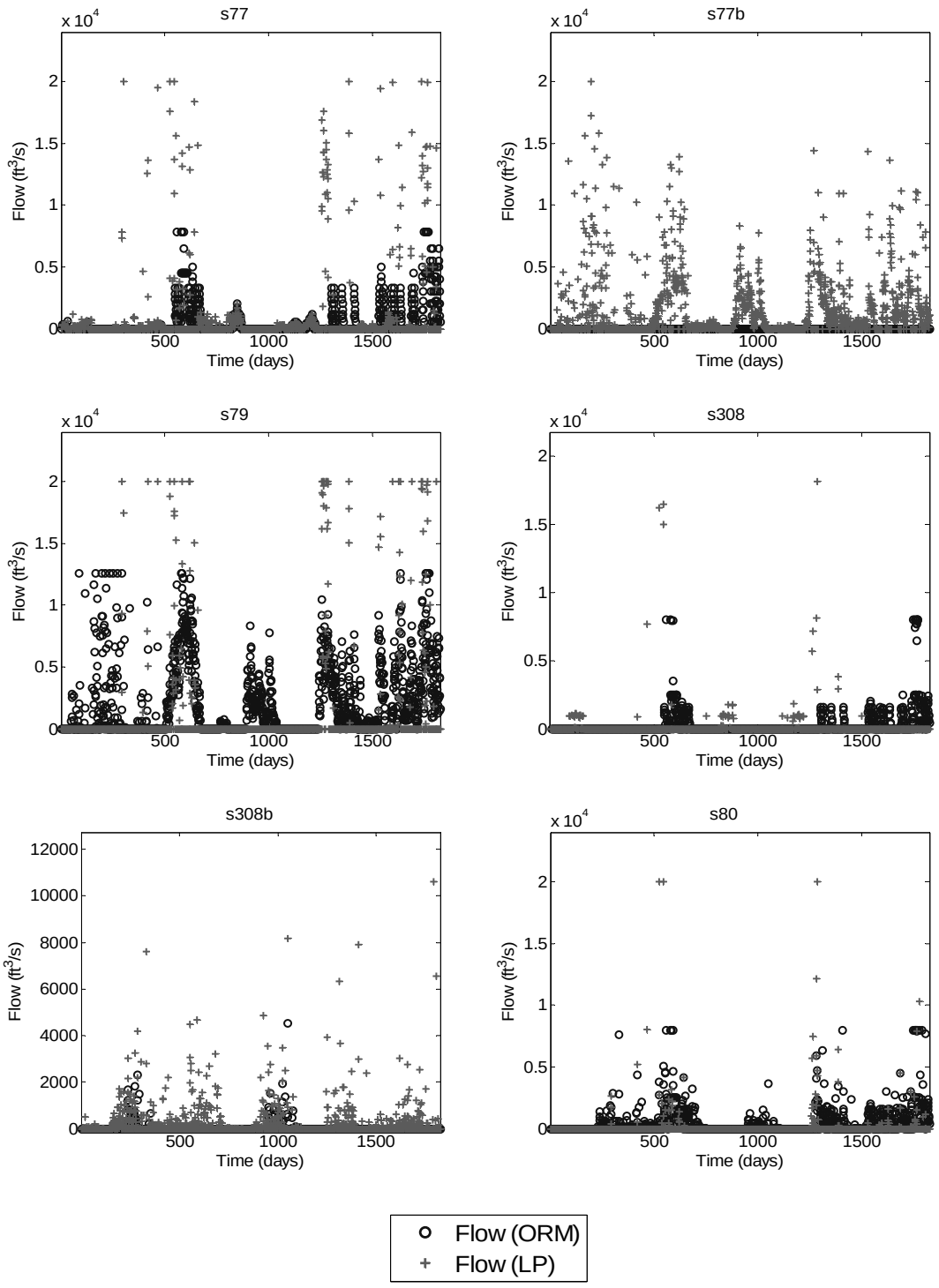


Figure A.4 Flow results for LP model containing no constraints on structure flows; that is, there are no flow capacity constraints, no management constraints, and no minimum flow constraints. ORM results are shown for comparison.

Figure A.4 (Continued)

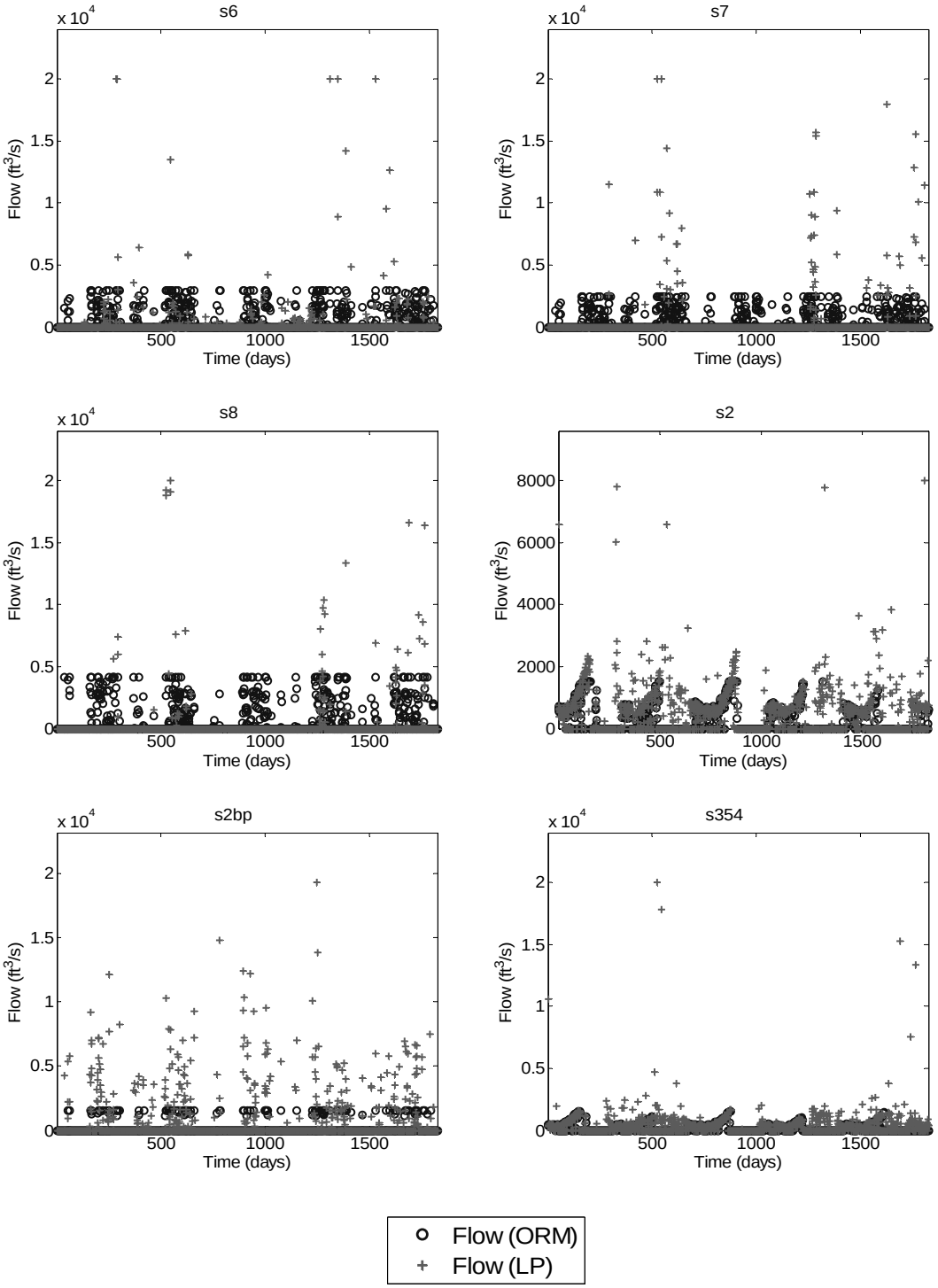


Figure A.4 (Continued)

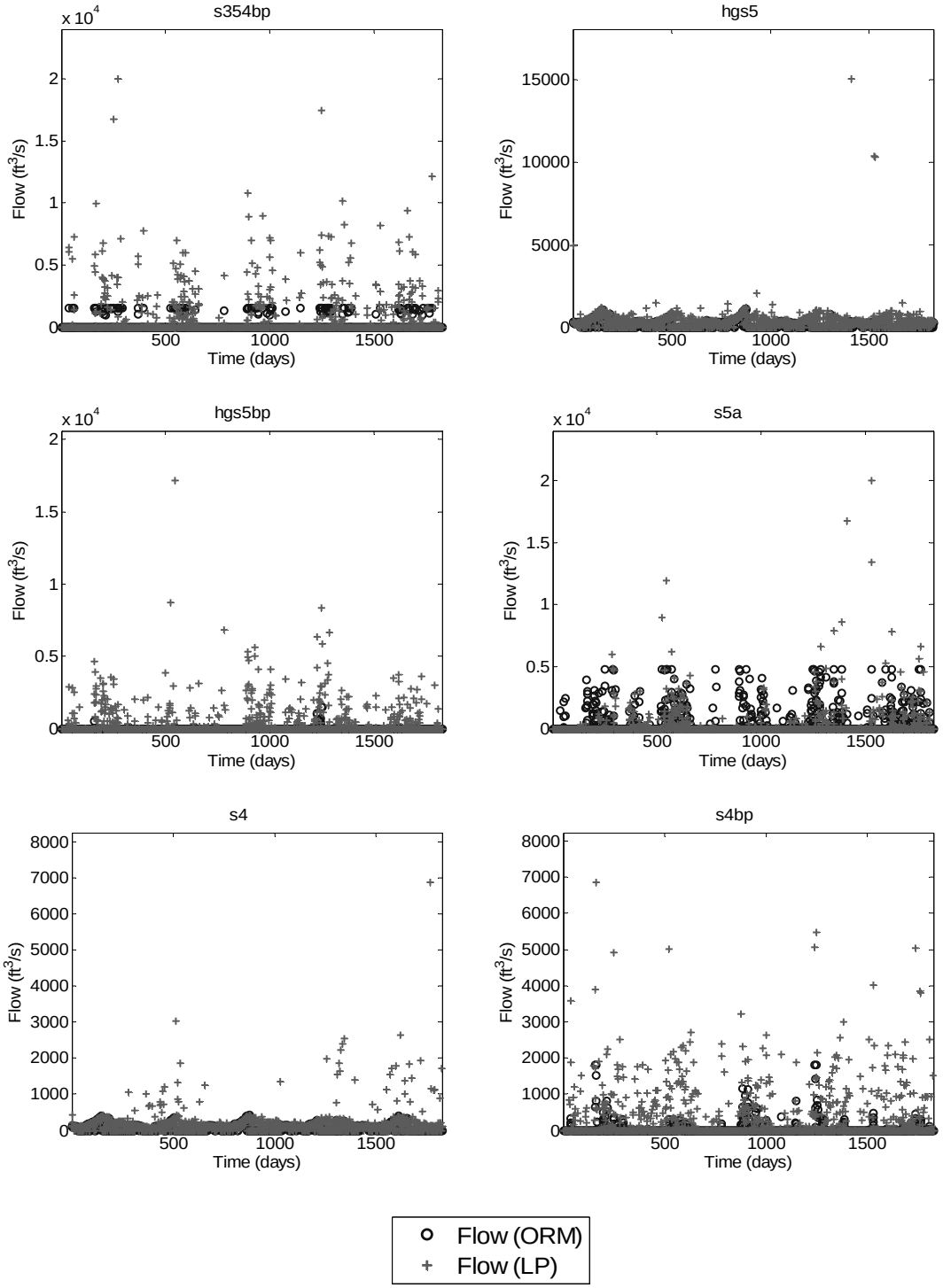


Figure A.4 (Continued)

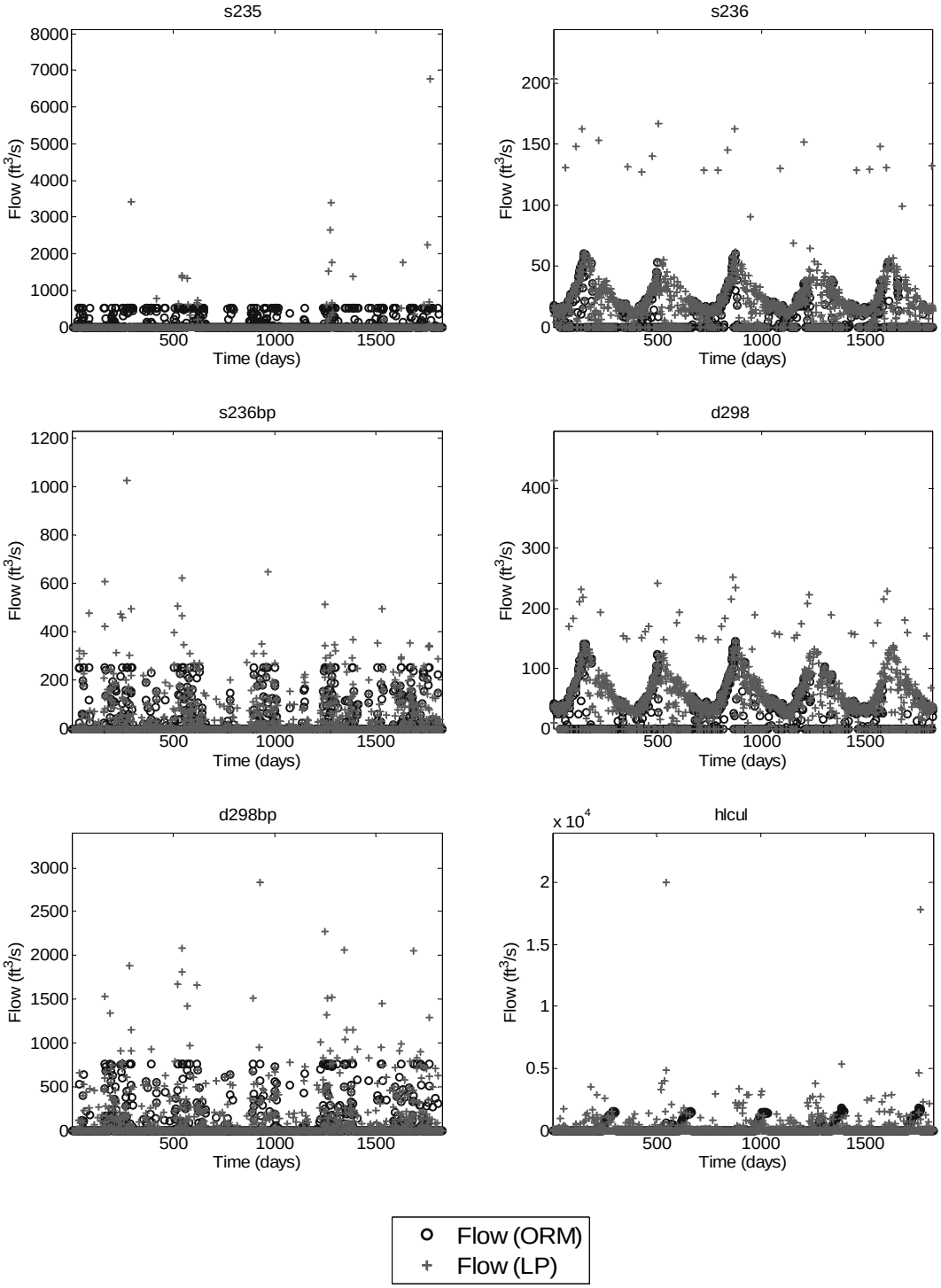


Figure A.4 (Continued)

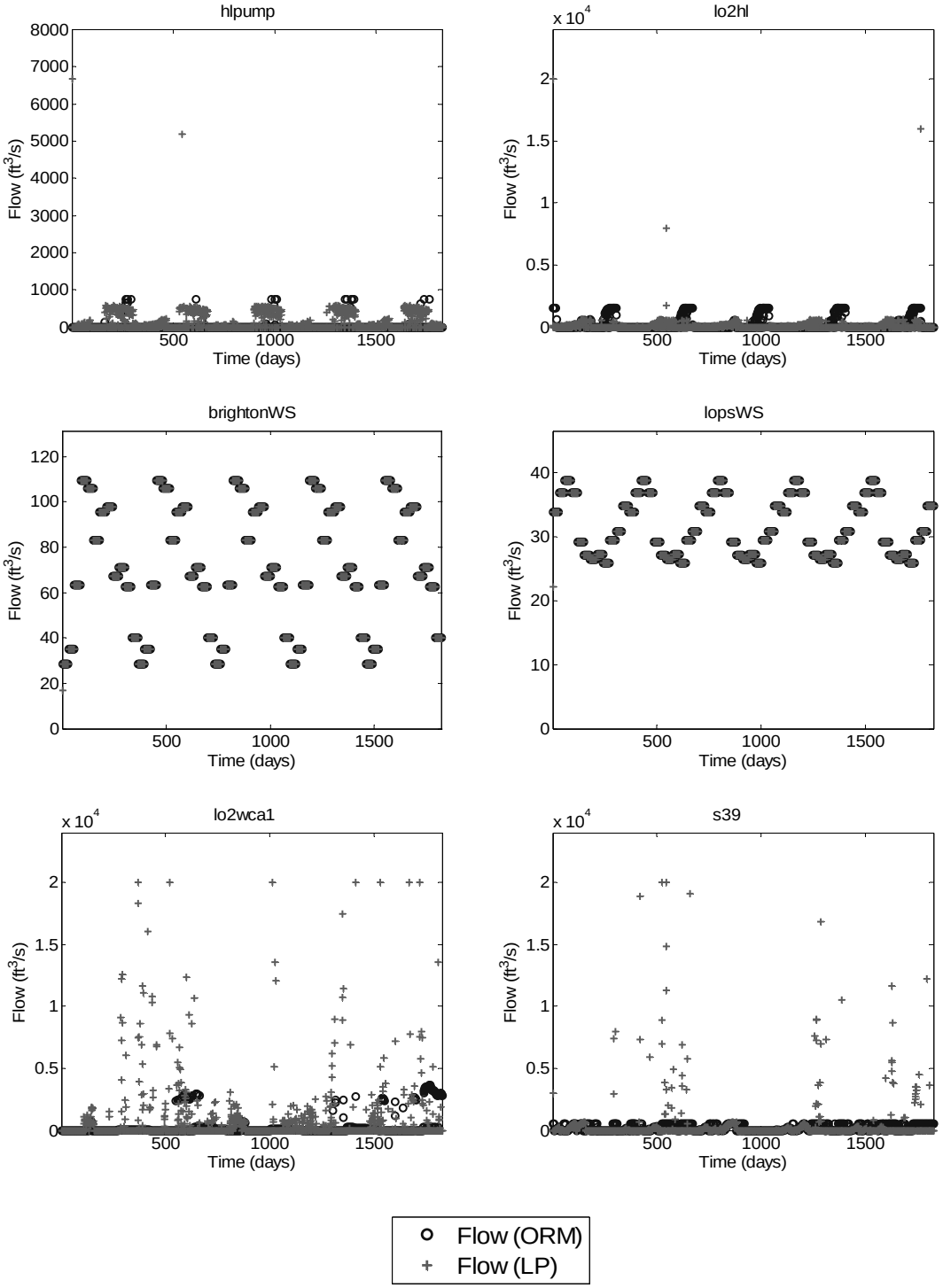


Figure A.4 (Continued)

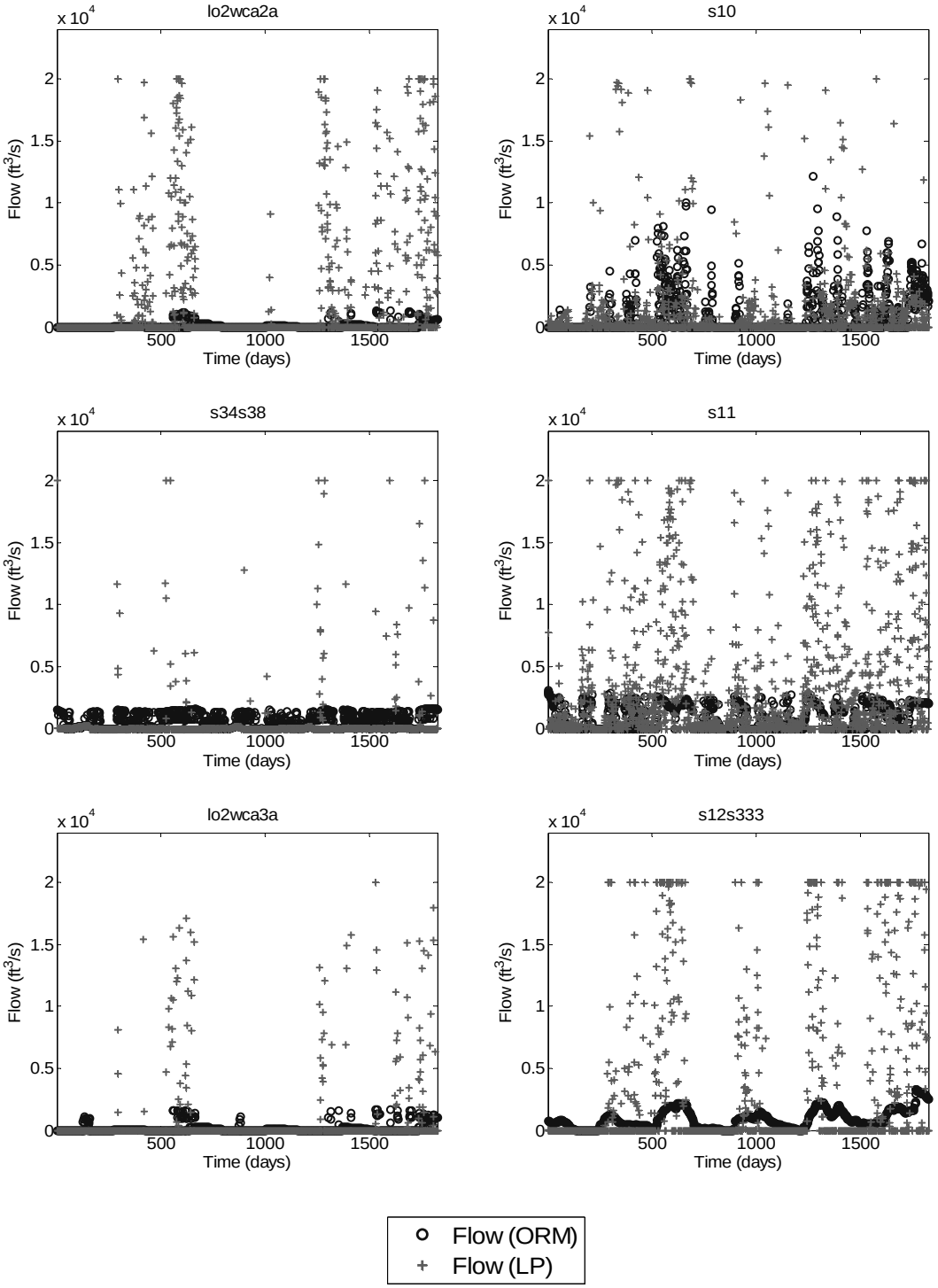
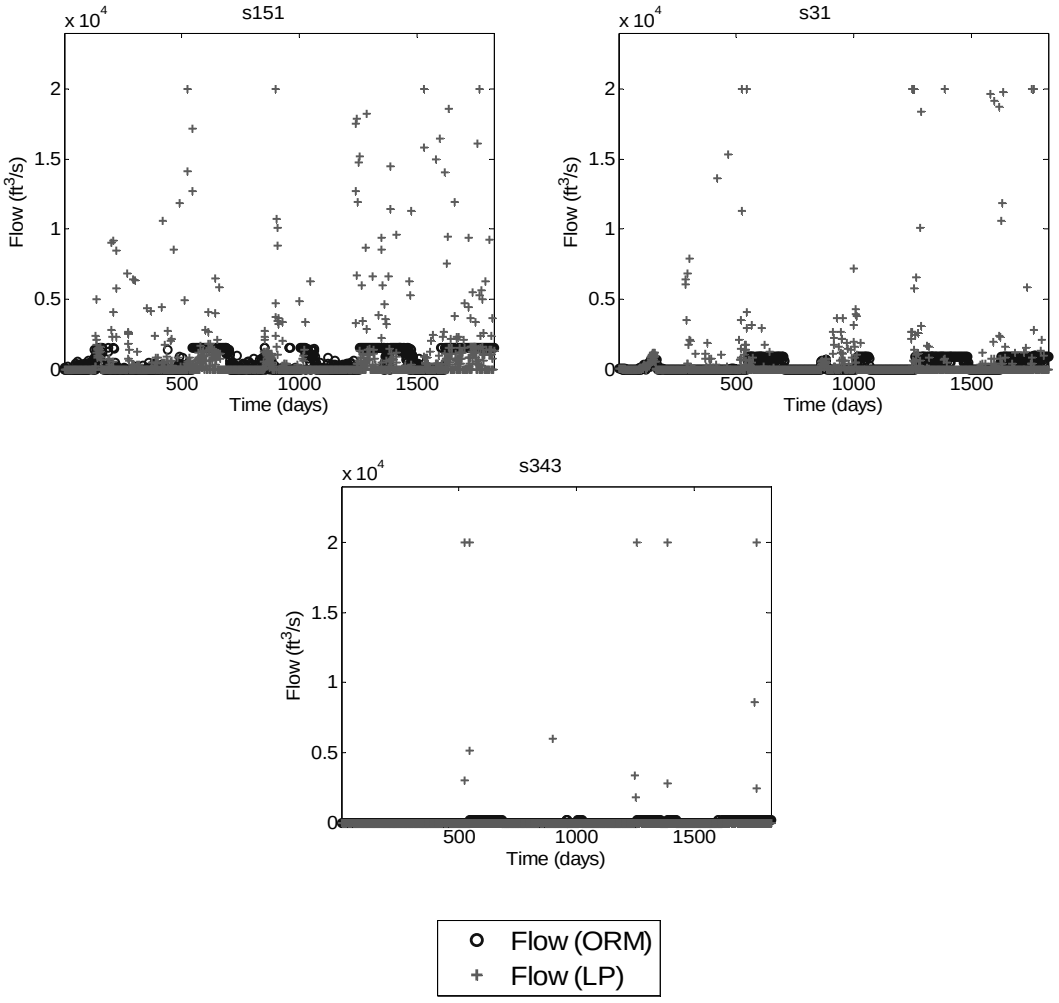


Figure A.4 (Continued)



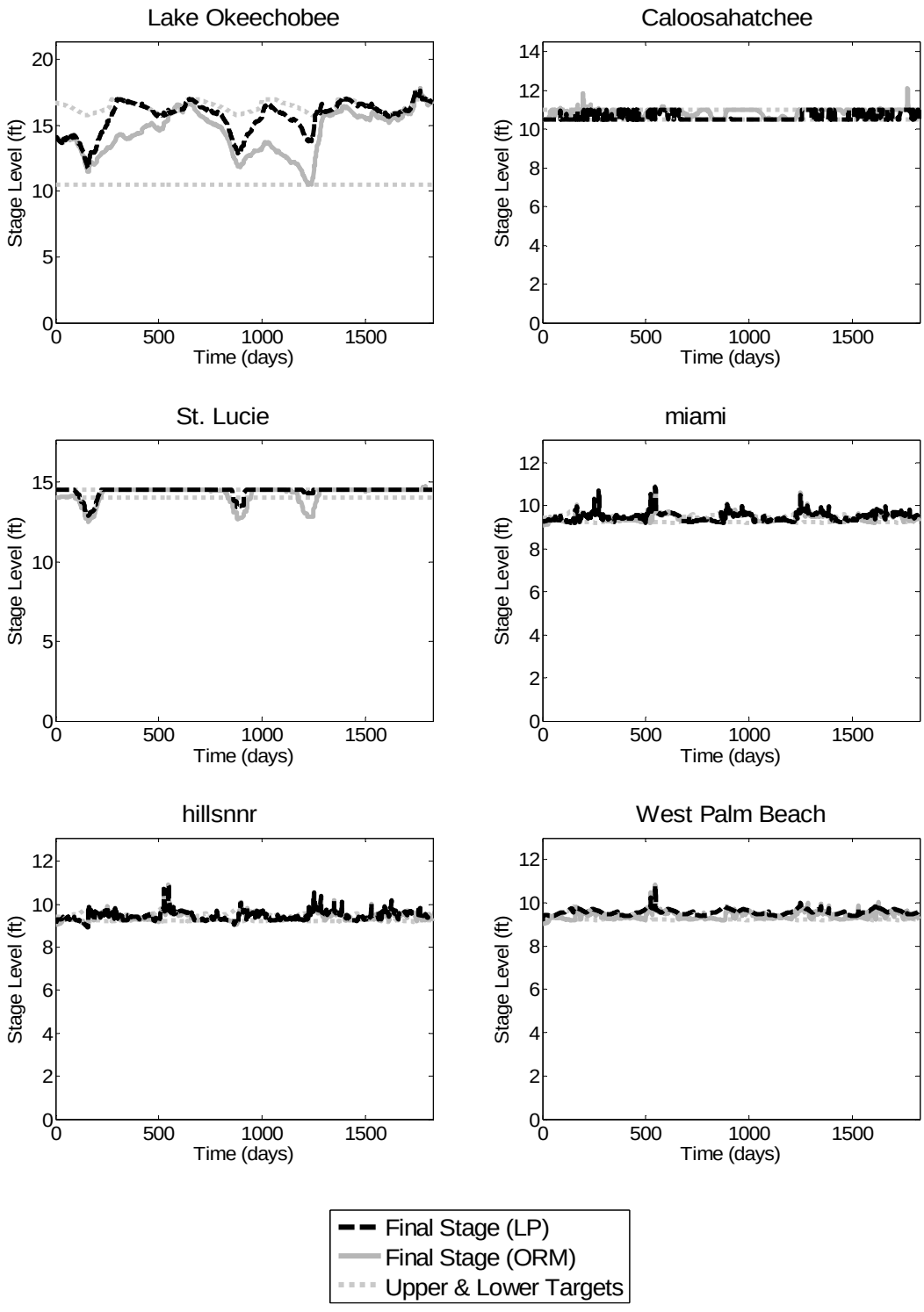


Figure A.5 Stage results for LP model containing flow capacity constraints. Management constraints and minimum flow constraints are excluded. ORM results shown for comparison.

Figure A.5 (Continued)

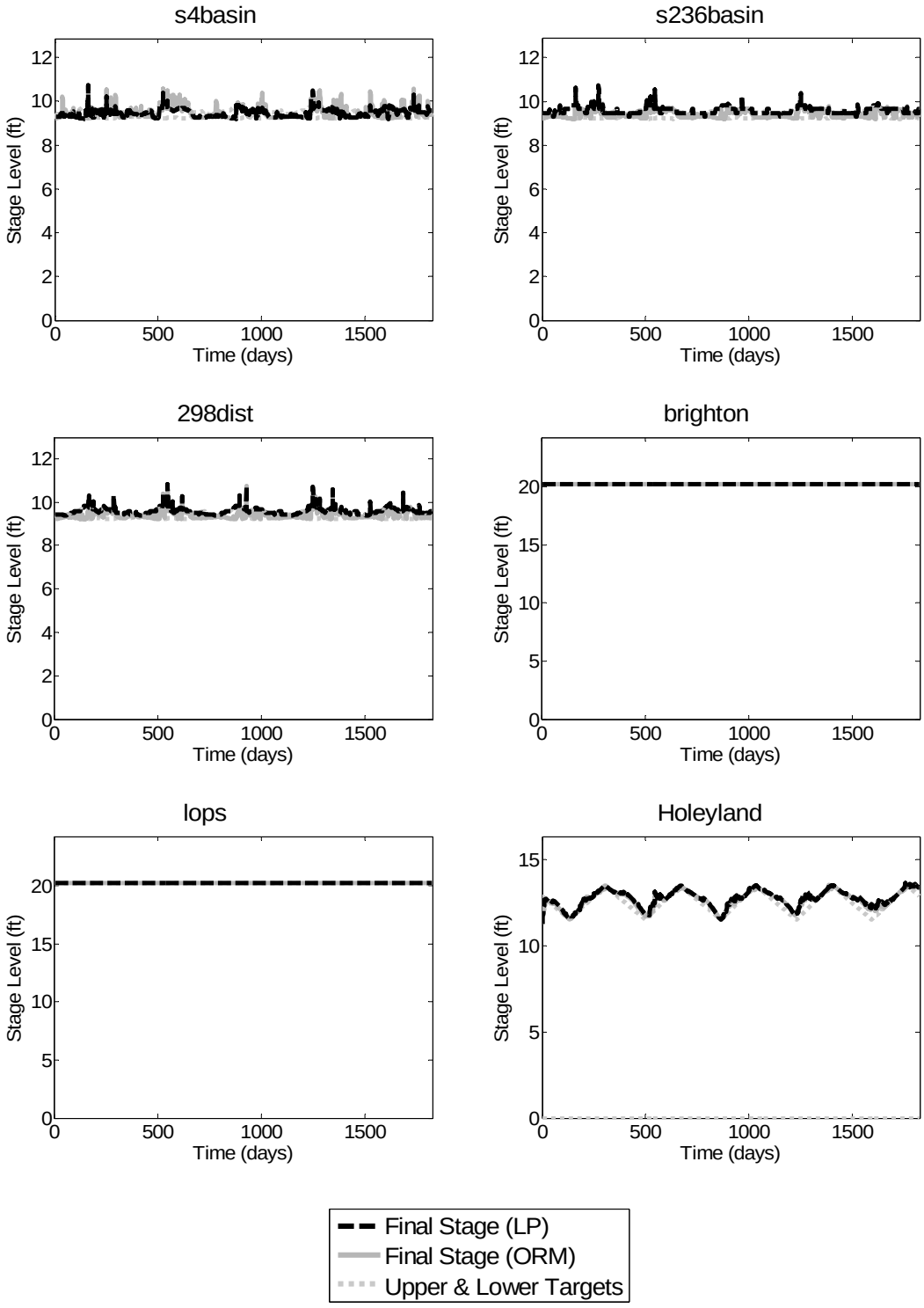


Figure A.5 (Continued)

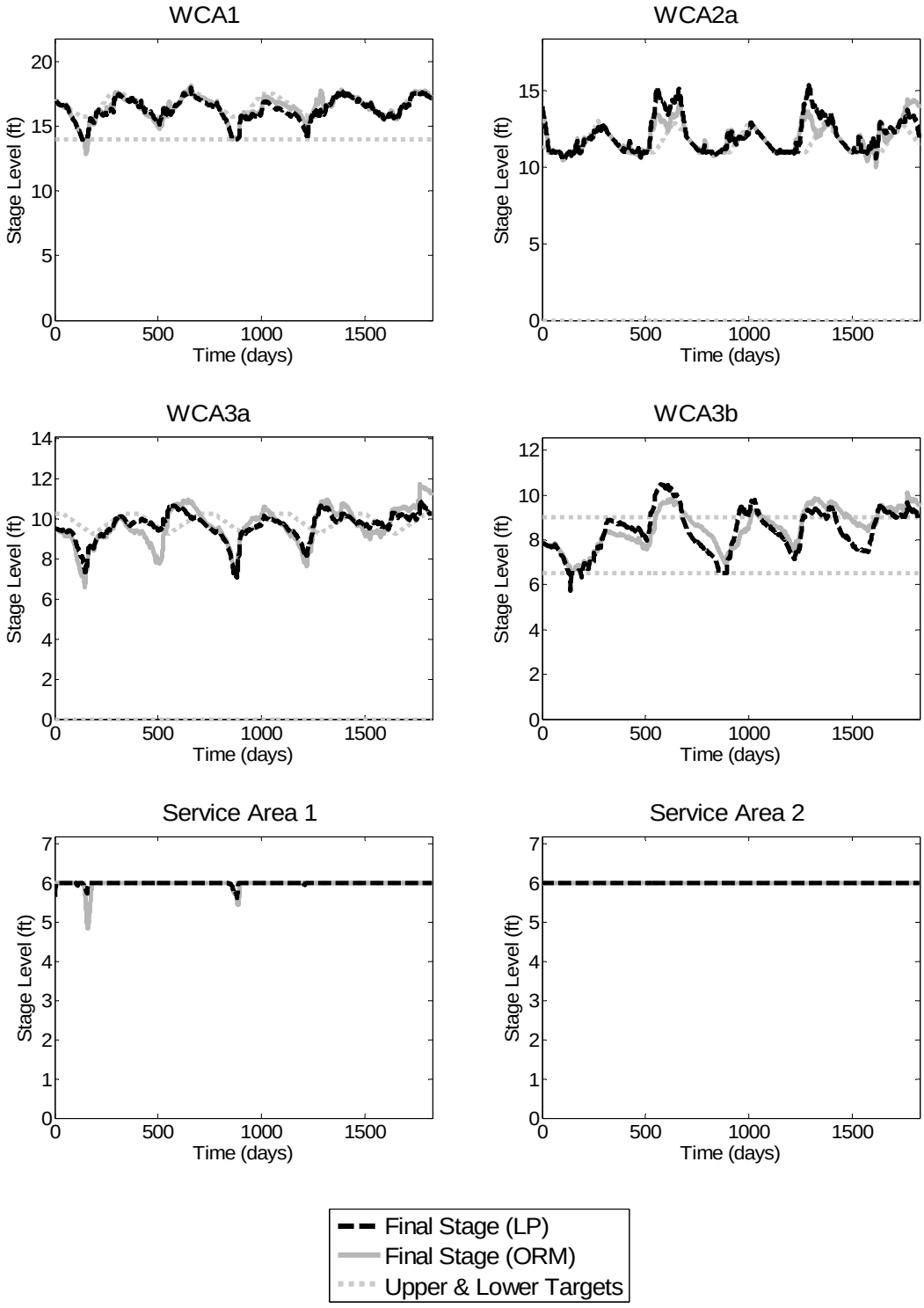
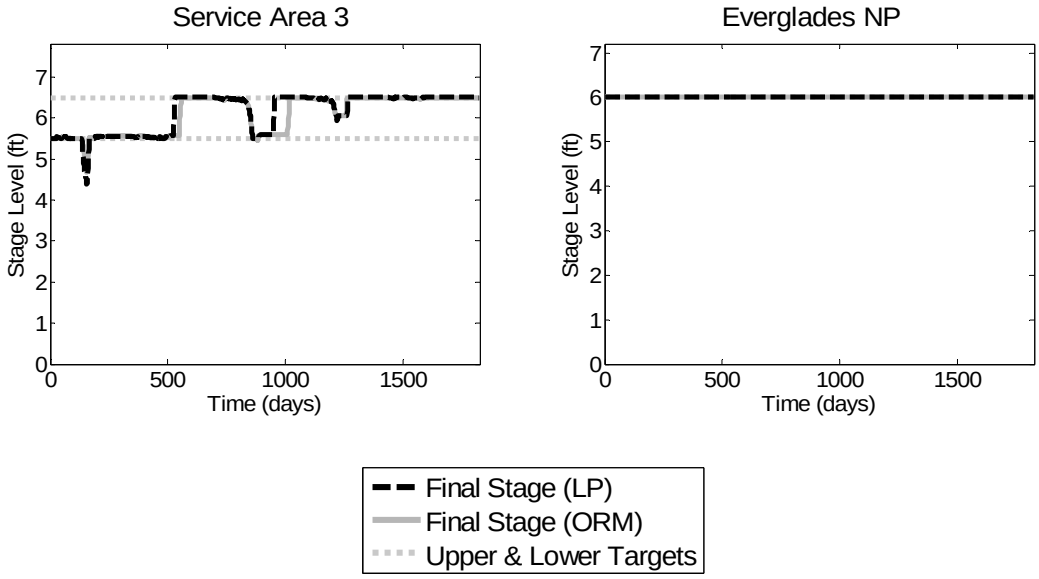


Figure A.5 (Continued)



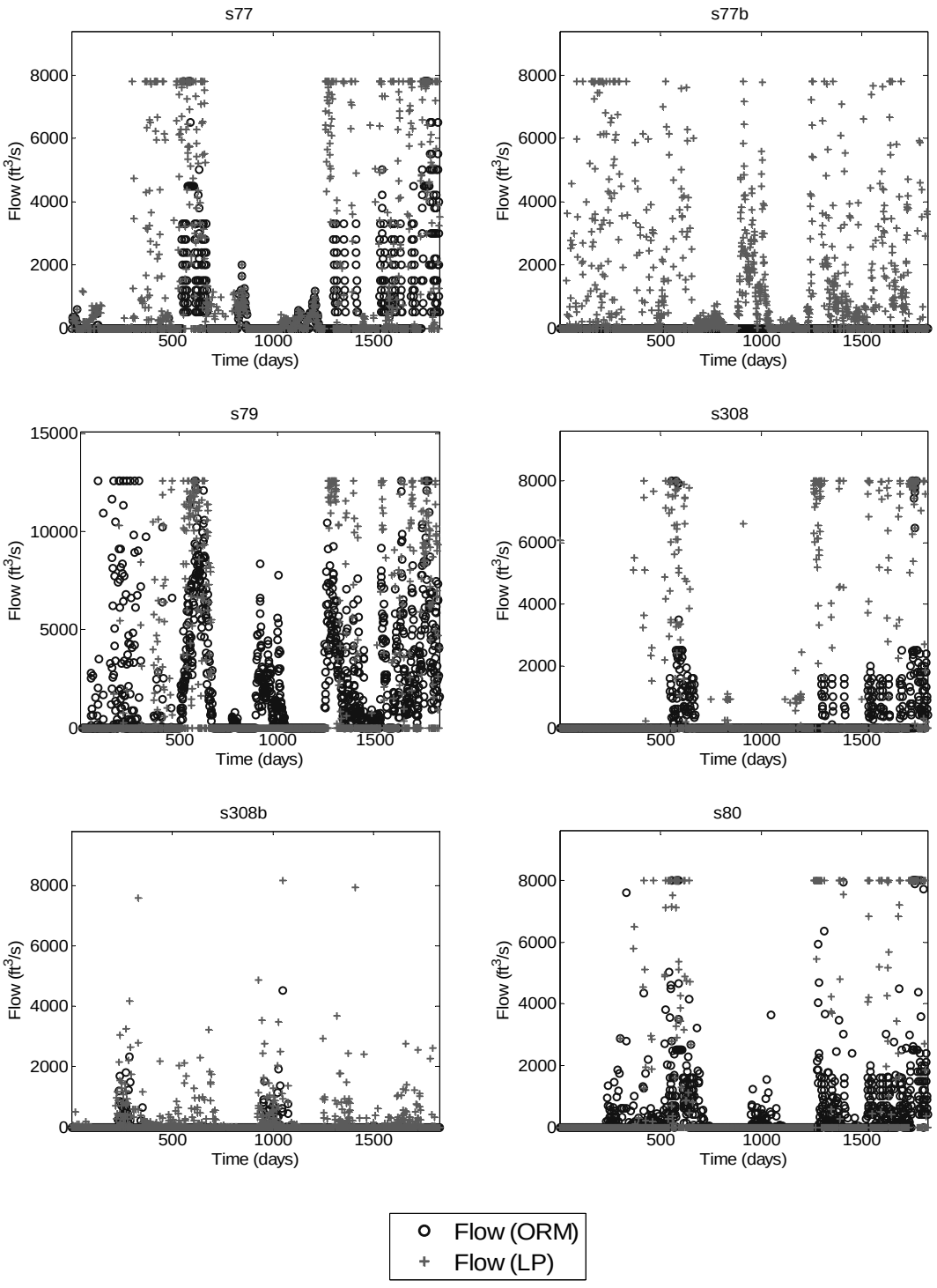


Figure A.6 Flow results for LP model containing flow capacity constraints. Management constraints and minimum flow constraints are excluded. ORM results shown for comparison.

Figure A.6 (Continued)

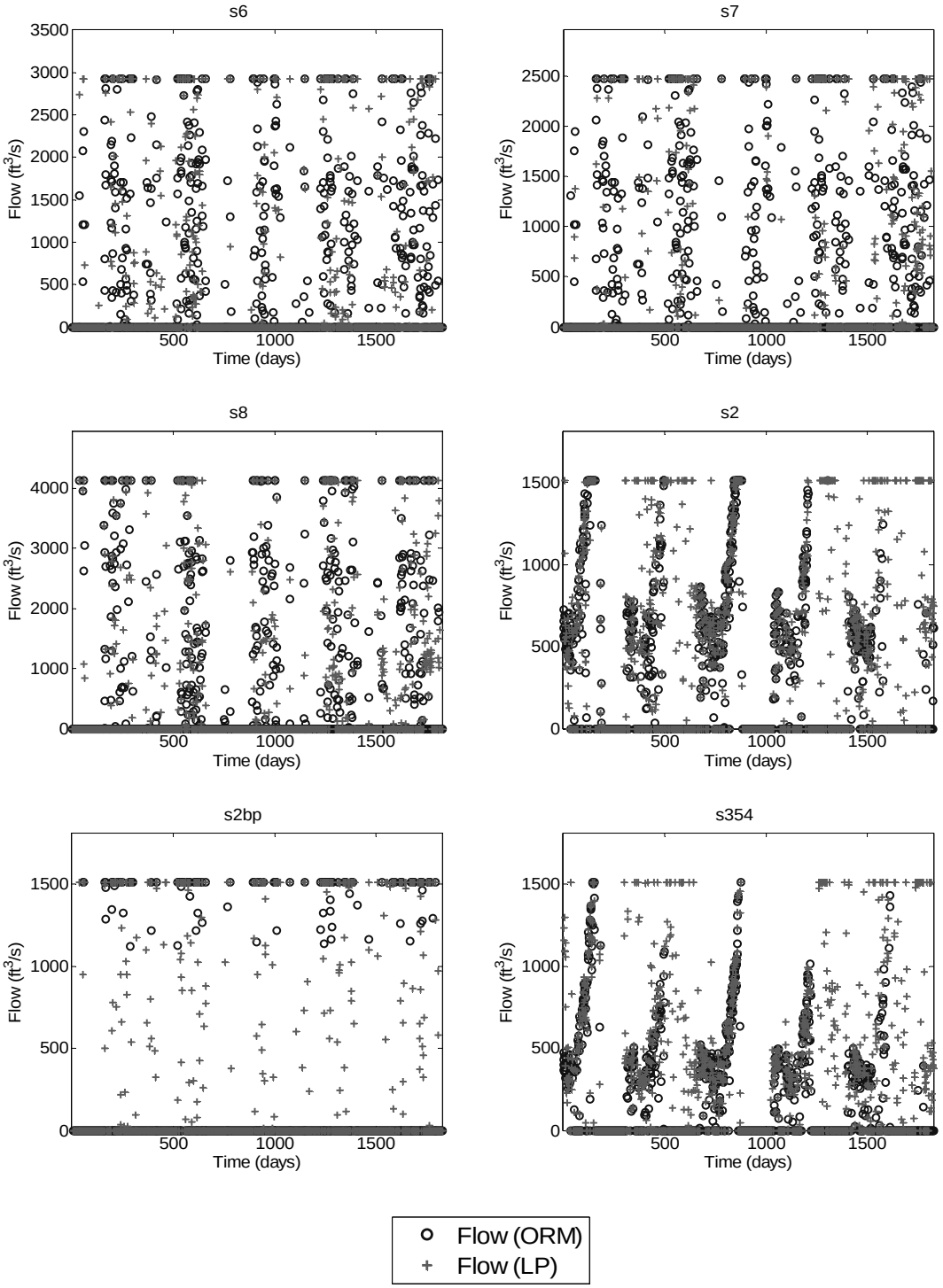


Figure A.6 (Continued)

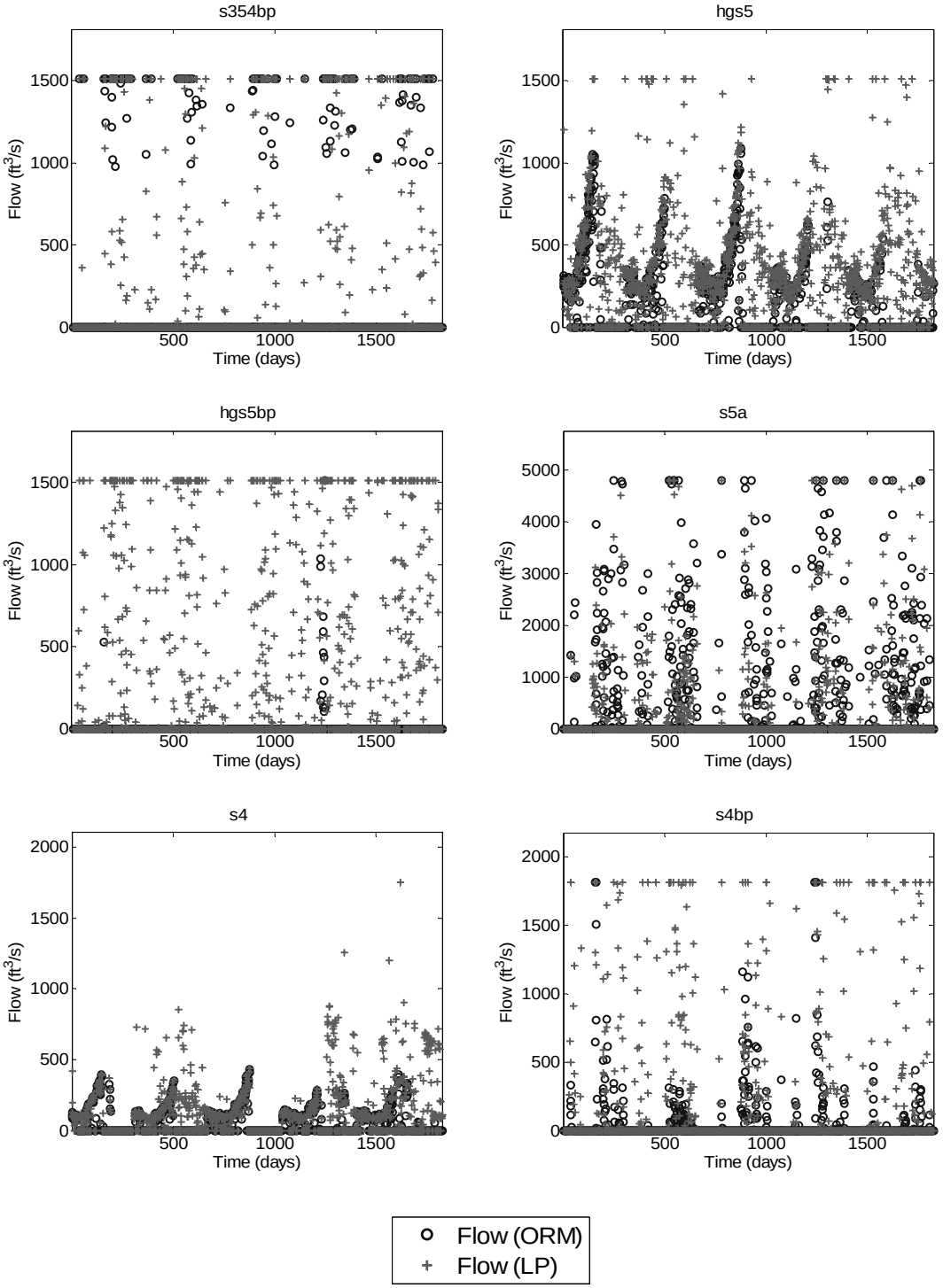


Figure A.6 (Continued)

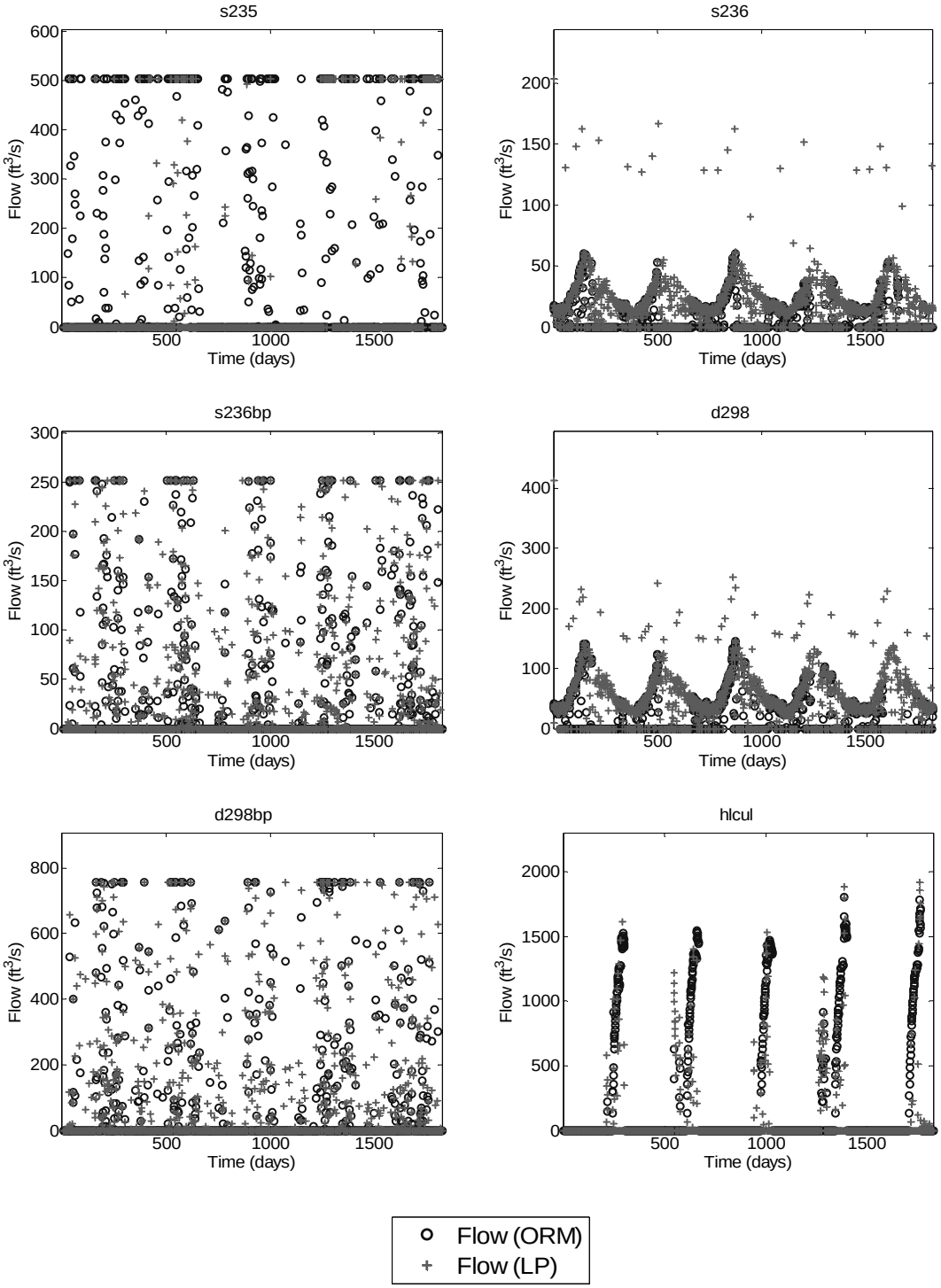


Figure A.6 (Continued)

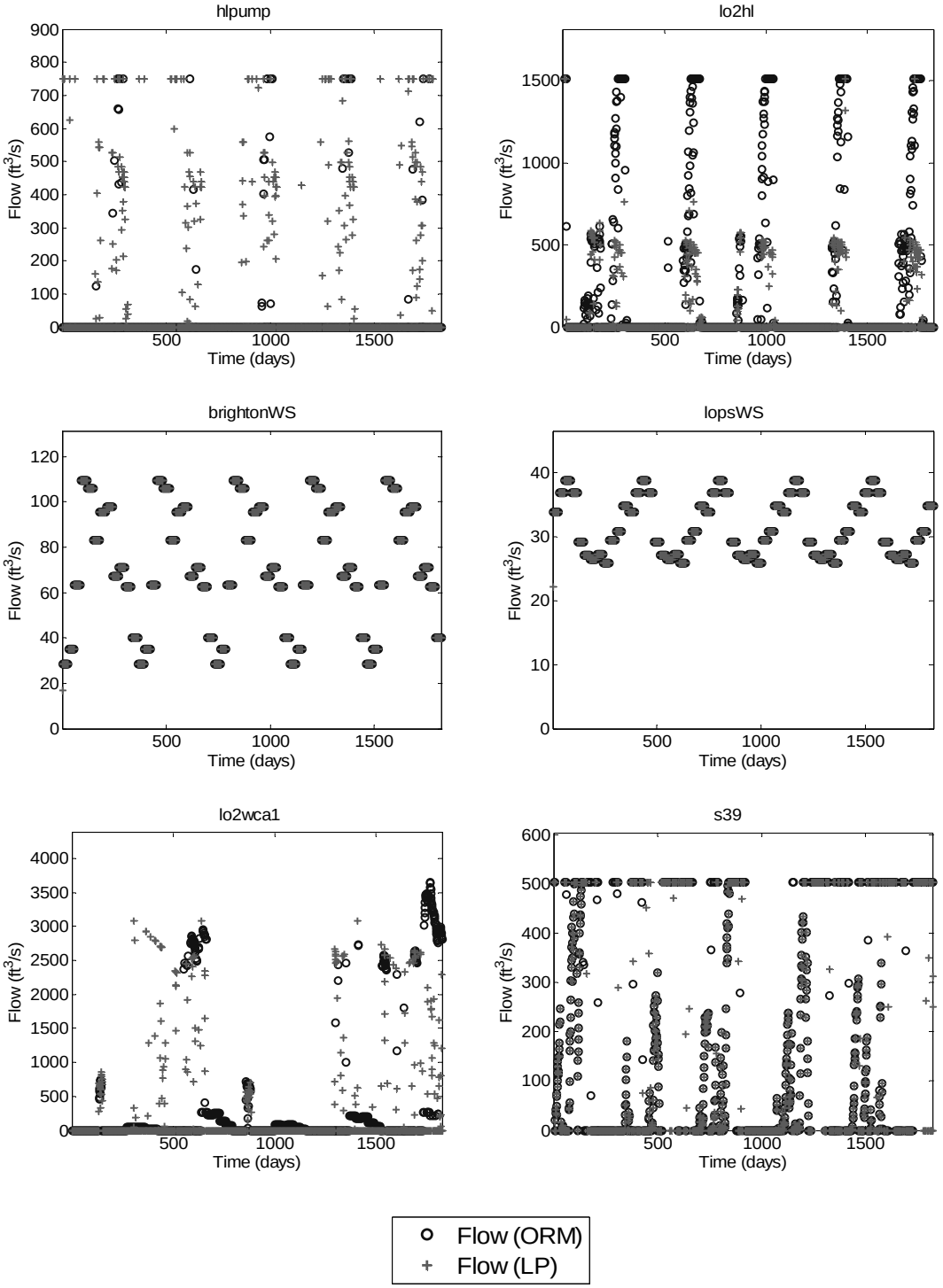


Figure A.6 (Continued)

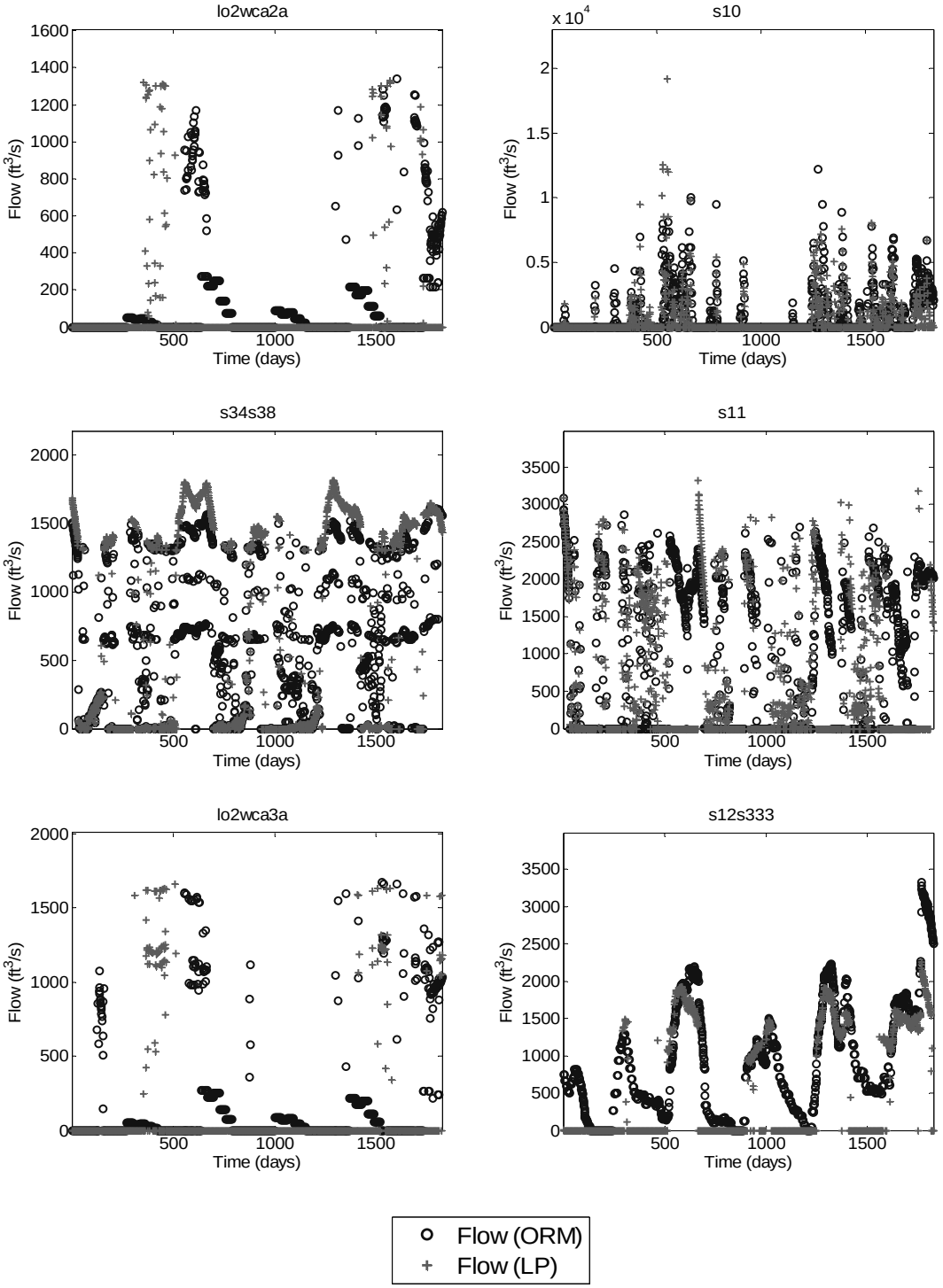
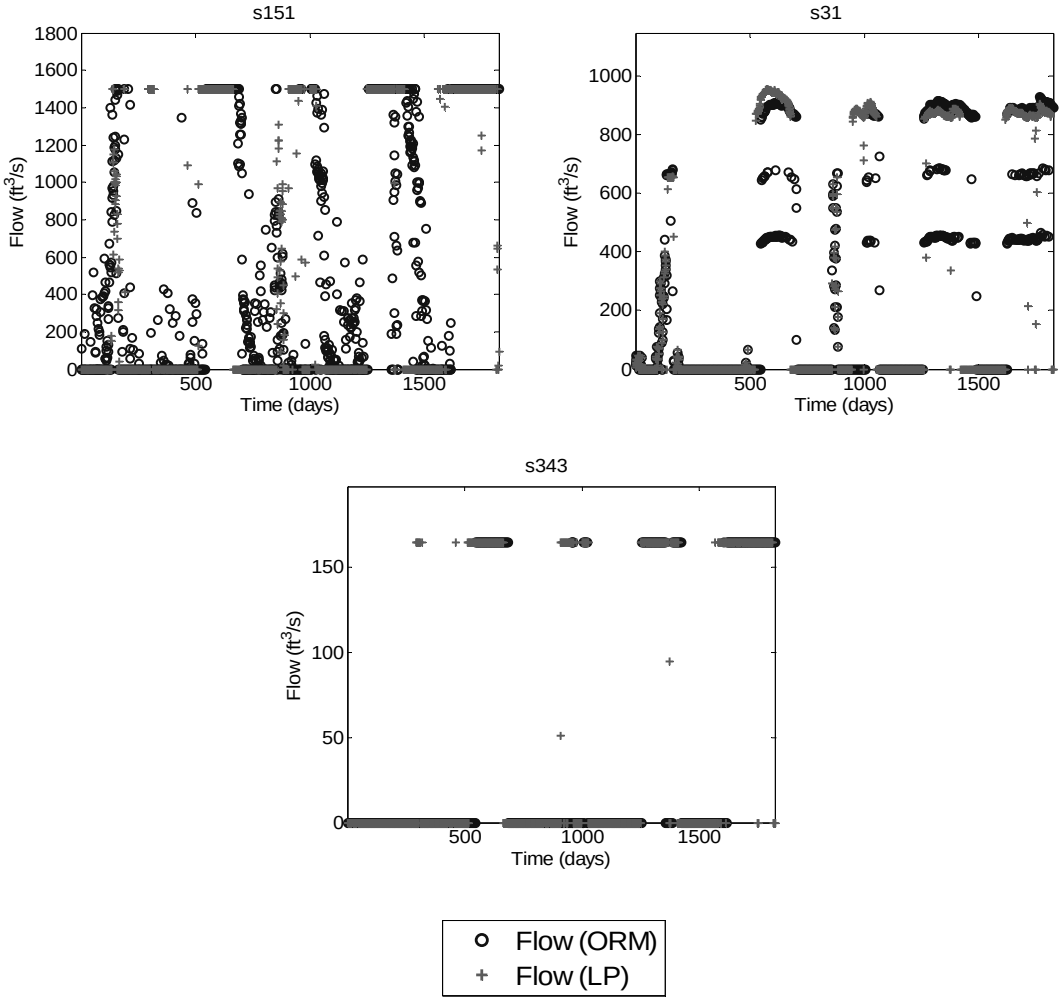


Figure A.6 (Continued)



APPENDIX B

RESULTS WITH ALTERNATE WEIGHTS

These figures show the results using an alternate set of weights on deviations from the basin targets. For these results, all of the weights are equal (and set to a value of one). The purpose of displaying these figures is to illustrate the importance of the weights in controlling the results of the LP model. An inappropriate set of weights could lead the LP model to perform poorly. It is therefore important that adequate care be taken to ensure that the chosen weights lead to acceptable results.

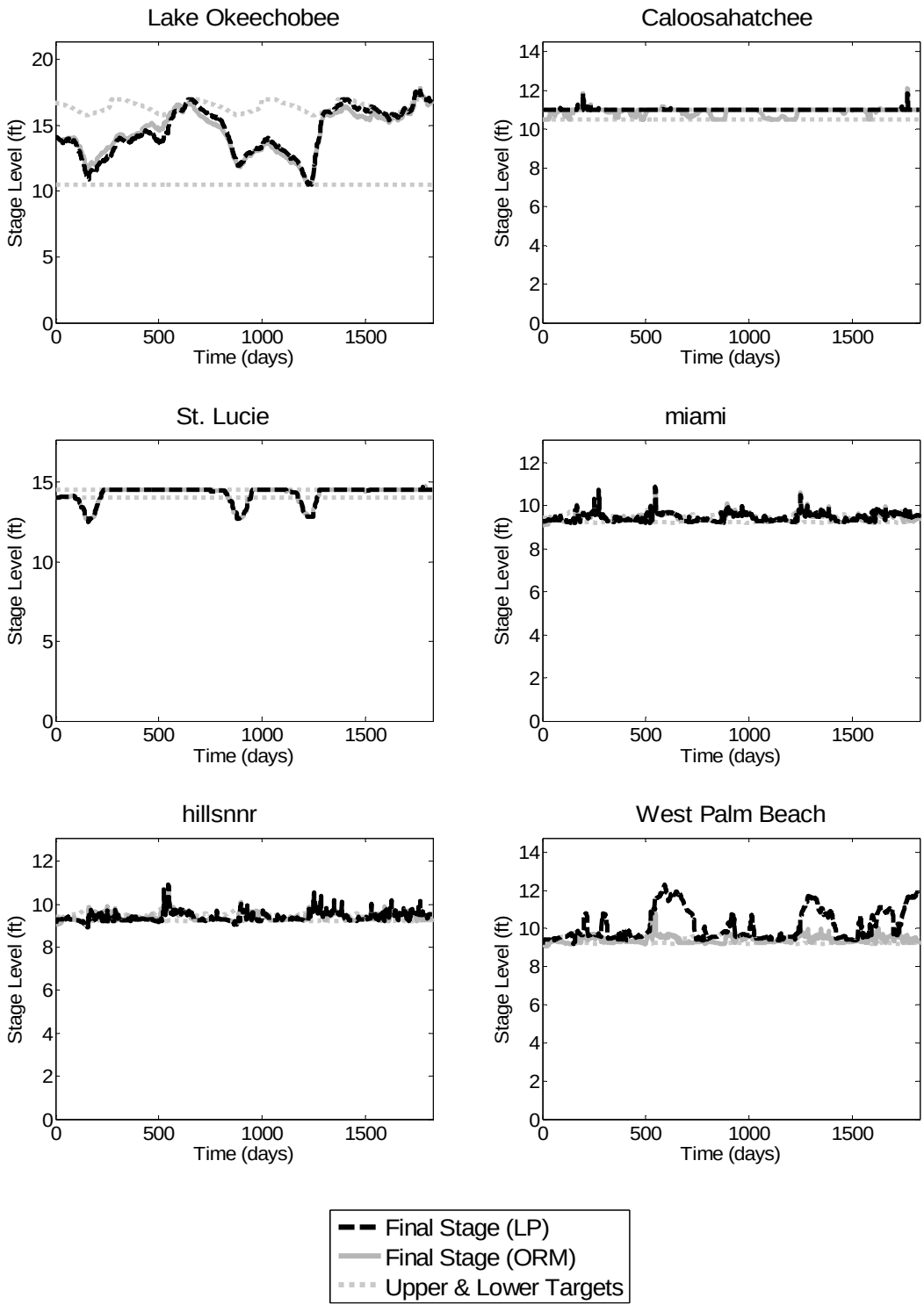


Figure B.1 Stage results for LP model and ORM using equal weights on all target deviations. This model contains all flow constraints.

Figure B.1 (Continued)

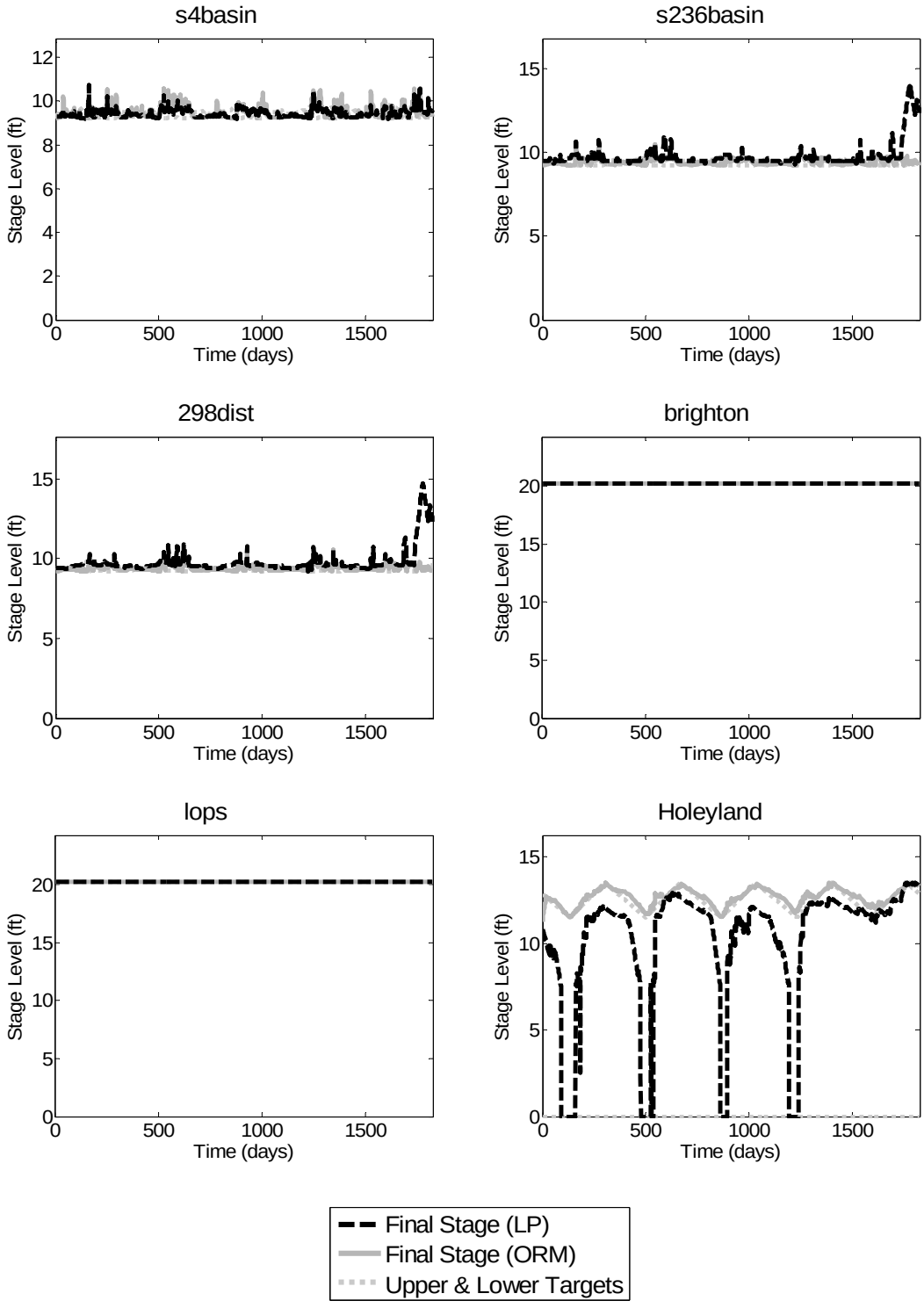


Figure B.1 (Continued)

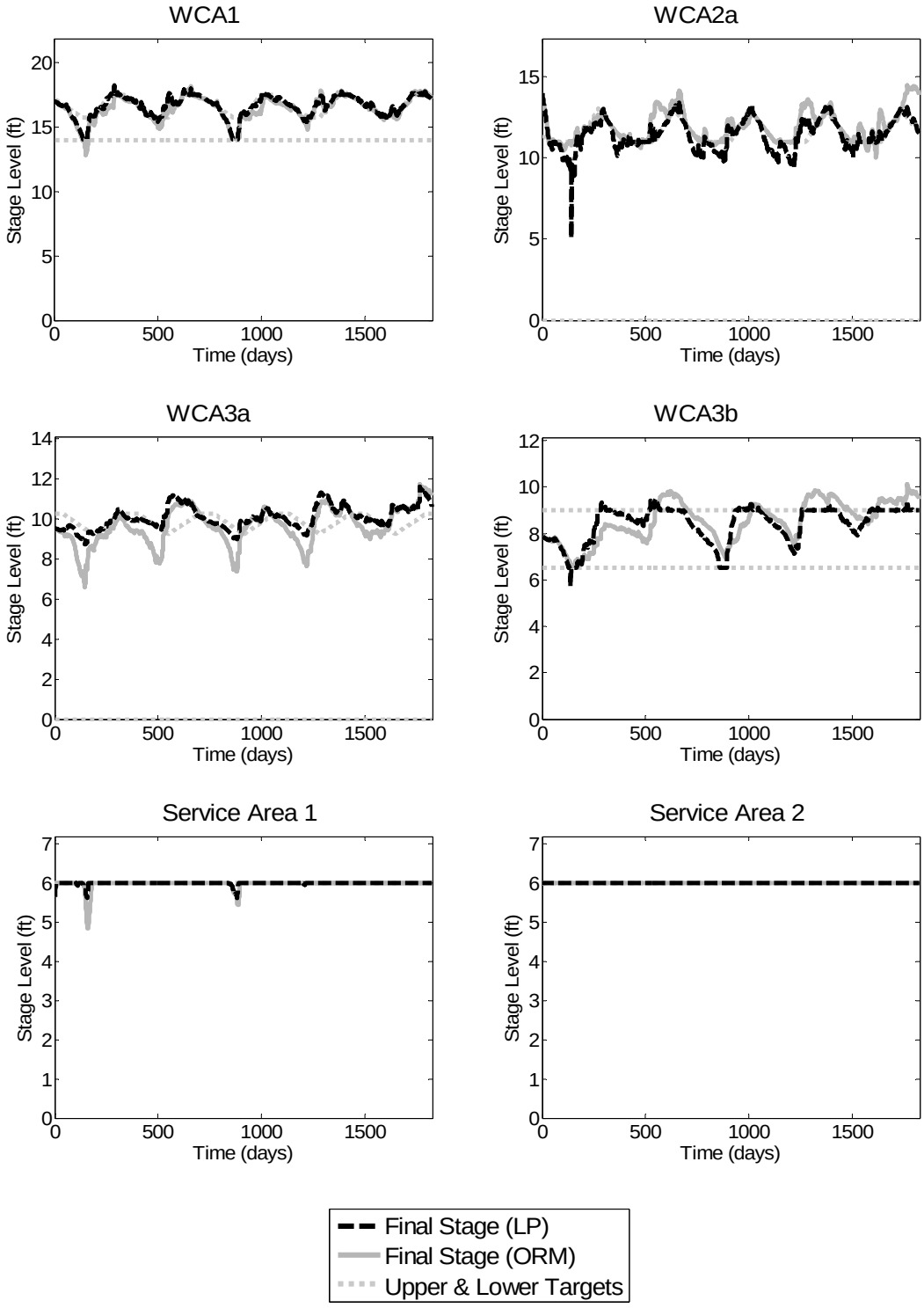
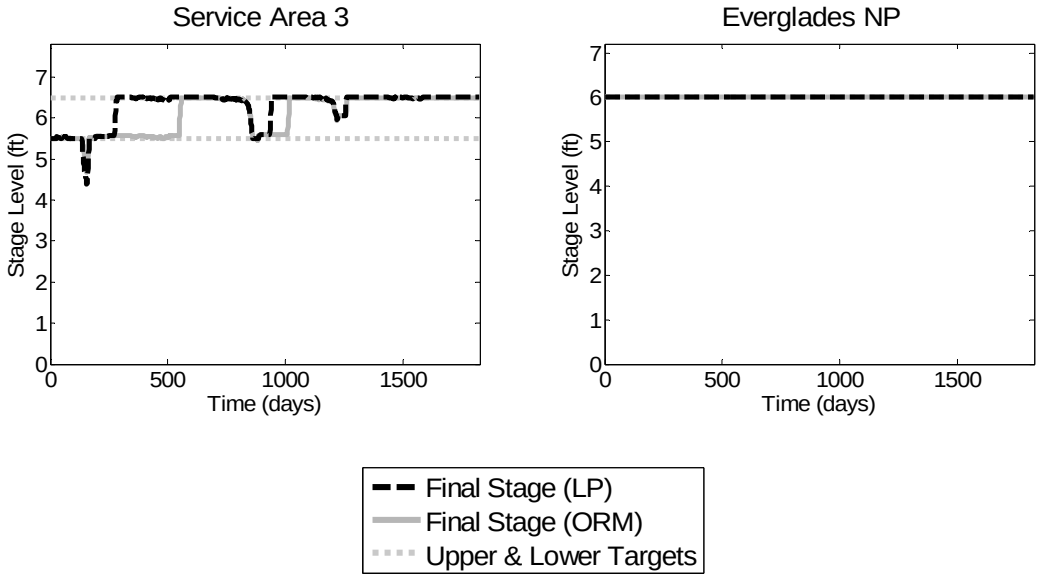


Figure B.1 (Continued)



APPENDIX C

LP MODEL AND SAMPLE INPUT FILE

This Appendix contains the LP model file, along with a sample input file for one time period. The LP model file forms the central component of the work discussed in this report. The model file contains the objective function and constraints, and initializes the parameters and variables to be used in solving the program. The model file also specifies the results to print. The input file defines the network and contains the values of the parameters for the time period being examined. The GLPK solver reads both of these files (the model file and the input file) and outputs the optimal results.

C.1. Linear Programming Model

```
set BASIN;
set STRUCTURE within BASIN cross BASIN;

/*-------------------------------------------------------------*/
/* general parameters                                          */
/*-------------------------------------------------------------*/

param dt         default 60*60*24;  /* number of seconds per day              */
param unitVol    default 43560/1E9; /* unit conversion ac-ft  --> 1E9 ft^3     */
param unitFlow   default dt/1E9;    /* unit conversion ft^3/s --> 1E9 ft^3/day */
param unitLength default 0.0833;    /* unit conversion inch   --> ft           */

param w_flowS333;
param w_minFlowDev;

/*-------------------------------------------------------------*/
/* BASIN parameters                                            */
/*-------------------------------------------------------------*/

param storageInit       { i in BASIN }, >=0;      /* 1E9 ft^3 */

/* STORAGE targets */
param basinTarget1      { i in BASIN }, >=0;      /* 1E9 ft^3 */
param basinTarget2      { i in BASIN }, >=0;      /* 1E9 ft^3 */

/* weights in objective function */
param w_target1Excess   { i in BASIN };
param w_target1Deficit  { i in BASIN };
param w_target2Deficit  { i in BASIN };
param w_demandDeficit   { i in BASIN };

param wsDemand          { i in BASIN }, >=0;      /* ac-ft    */
param rainfall          { i in BASIN }, >=0;      /* 1E9 ft^3 */

/* runoff can be negative, since some of the runoff data is < 0 */
```

```
param runoff            { i in BASIN };           /* ac-ft    */

param evap              { i in BASIN }, >=0;      /* 1E9 ft^3 */
param deltaStorage      { i in BASIN };           /* ac-ft    */

/*-------------------------------------------------------------*/
/* STRUCTURE parameters                                        */
/*-------------------------------------------------------------*/

param name              { (i,j) in STRUCTURE }, symbolic;
param designCap         { (i,j) in STRUCTURE }, >=0;    /* ft^3/s */
param maxFlow           { (i,j) in STRUCTURE }, >=0;    /* ft^3/s */
param manCon            { (i,j) in STRUCTURE }, >=0;    /* ft^3/s */
param useManCon         { (i,j) in STRUCTURE };         /* binary */
param minflow           { (i,j) in STRUCTURE }, >=0;    /* ft^3/s */
param w_flow            { (i,j) in STRUCTURE };

/*-------------------------------------------------------------*/
/* Variables                                                   */
/*-------------------------------------------------------------*/

var flow                { (i,j) in STRUCTURE }, >=0, <= designCap[i,j]*unitFlow;
                                                        /* 1E9 ft^3/day  */

var storageFinal        { i in BASIN }, >=0;            /* 1E9 ft^3      */

var target1Excess       { i in BASIN }, >=0;            /* 1E9 ft^3      */
var target1Deficit      { i in BASIN }, >=0;            /* 1E9 ft^3      */
var target2Deficit      { i in BASIN }, >=0;            /* 1E9 ft^3      */
var demandDeficit       { i in BASIN }, >=0;            /* ac-ft         */

var wsDemandActual      { i in BASIN }, >= 0;           /* ac-ft         */

var flowS333, >=0;                                      /* 1E9 ft^3/day   */
var flowS12, >=0;                                       /* 1E9 ft^3/day   */
var minFlowDev          { (i,j) in STRUCTURE }, >=0;    /* 1E9 ft^3/day   */

/*-------------------------------------------------------------*/
/* Model objective: Minimize total weighted target deviations  */
/*-------------------------------------------------------------*/

minimize objective :

sum { i in BASIN } ( w_target1Excess[i]  * target1Excess[i]
                   + w_target1Deficit[i] * target1Deficit[i]
                   + w_target2Deficit[i] * target2Deficit[i]
                   + w_demandDeficit[i]  * demandDeficit[i] * unitVol )

/* flow penalties to enforce coordinators */
                   + sum { (i,j) in STRUCTURE } ( w_flow[i,j] * flow[i,j] )

/* for S12S333 flow */
                    -   ( w_flowS333*flowS333 )

/* to enforce minimum flow constraint */
                    + sum { (i,j) in STRUCTURE } w_minFlowDev * minFlowDev[i,j];

/*-------------------------------------------------------------*/
/* Model constraints                                           */
/*-------------------------------------------------------------*/

/**  Determine deviations  **/
subject to Target1Deviation { i in BASIN } :
       storageFinal[i] = basinTarget1[i] + target1Excess[i] – target1Deficit[i];
subject to Target2Deviation { i in BASIN } :
       storageFinal[i] >= basinTarget2[i] – target2Deficit[i];

subject to DemandDeviation { i in BASIN } :
       wsDemandActual[i] = wsDemand[i] – demandDeficit[i];
```

```
/**  Continuity constraints **/
subject to MassConservation { i in BASIN } :
       storageFinal[i] = storageInit[i] + runoff[i]*unitVol + rainfall[i] – evap[i]
                              - wsDemandActual[i]*unitVol + deltaStorage[i]*unitVol
                              + sum{ (j,i) in STRUCTURE } flow[j,i]
                              -   sum{ (i,j) in STRUCTURE } flow[i,j];

/**  Flow Constraints  **/
subject to FlowCapacity { (i,j) in STRUCTURE } :
       flow[i,j] <= maxFlow[i,j]*unitFlow;

subject to ManagedConstraint { (i,j) in STRUCTURE } :
       if useManCon[i,j]
       then flow[i,j] <= manCon[i,j]*unitFlow;
subject to MinimumFlow { (i,j) in STRUCTURE } :
       minFlowDev[i,j] >= minflow[i,j]*unitFlow – flow[i,j];

/**  S12S33 CONSTRAINTS  **/
subject to Flow1_s12s333 :
       flow['wca3a','enp'] = flowS12 + flowS333;
subject to Flow2_s12s333 :
       flowS333 <= 0.55*flow['wca3a','enp'];

solve;


/** print output file **/
printf{1..65} "="; printf "\n\n";
printf "        BASIN     Demand  DemandAct  storInit      storFin         bT1
bT2 w_t1E w_t1D w_t2D\n";

printf "## units            ACFT       ACFT   1E9ft^3      1E9ft^3     1E9ft^3
1E9ft^3\n";

printf{ i in BASIN}: " %12s %11.4f %10.4f %10.4f %13.8f %12.4f %8.4f\n",
i,  wsDemand[i], wsDemandActual[i],
storageInit[i], storageFinal[i], basinTarget1[i], basinTarget2[i];

printf{1..65} "="; printf "\n";

printf "         i         j         name         flow     maxFlow   designCap
manCon   useManCon\n";

printf "         -         -         ----         ----     -------   ---------      --
----   ---------\n";

printf "## units                                ft^3/s      ft^3/s      ft^3/s
ft^3/s\n";

printf{ (i,j) in STRUCTURE }: "%10s %9s %12s %12.4f %11.4f %11.4f %11.4f %11d\n",
    i, j, name[i,j], (flow[i,j]/unitFlow), maxFlow[i,j], designCap[i,j], manCon[i,j],
useManCon[i,j];

printf{1..65} "="; printf "\n";

end;
```

C.2. Sample LP Input File

```
/*============================================================

  Input data for the LP model for the time period: 31-Dec-1969

 ============================================================*/


/*-------------------------------------------------------------*/
/* general parameters                                          */
/*-------------------------------------------------------------*/


 param  w_flowS333        :=             1.000 ;
 param  w_minFlowDev      :=          1000.000 ;


/*-------------------------------------------------------------*/
/* BASIN parameters                                            */
/*-------------------------------------------------------------*/


 param :
 BASIN :        storageInit    basinTarget1    basinTarget2 :=
 /* unit          1E9 ft^3        1E9 ft^3        1E9 ft^3   */
 lo            207.65000000          207.50           96.00
 caloosa        11.64000000           11.64           11.11
 stlucie        15.34000000           15.34           14.82
 miami           4.36505984            4.38            3.85
 hillsnnr        6.66723805            6.71            5.89
 wpb             2.89997225            2.90            2.55
 s4basin         1.17997208            1.18            1.04
 s236basin       0.16997390            0.17            0.15
 d298dist        0.38997222            0.39            0.34
 brighton        6.96000000            6.96            6.96
 lops            6.96000000            6.96            6.96
 holeyland       4.11110350            0.00            3.36
 wca1           14.10788264           14.15            1.18
 wca2a           8.48040391            2.87            0.00
 wca3a          39.62468910           39.64            0.00
 wca3b           8.76051410            9.08            0.44
 sa1             3.68000000            3.68            3.68
 sa2             3.68000000            3.68            3.68
 sa3             3.98425008            3.99            3.37
 bcnp            0.00000000            0.00            0.00
 enp             4.55000000            4.55            4.55
 ocean         100.00000000            0.00            0.00  ;


 param :      w_target1Deficit w_target1Excess  w_target2Deficit w_demandDeficit :=
 lo                    1.1000            0.00              0.50         1000.00
 caloosa               0.5000            0.50              2.00         1000.00
 stlucie               1.1000            1.00              1.10         1000.00
 miami                 1.0000            0.90              1.10         1000.00
 hillsnnr              1.0000            0.90              1.10         1000.00
 wpb                   1.0000            0.90              1.10         1000.00
 s4basin               1.0000            0.90              1.10         1000.00
 s236basin             1.0000            0.90              1.10         1000.00
 d298dist              1.0000            0.90              1.10         1000.00
 brighton              1.0000            1.00              2.00         1000.00
 lops                  1.0000            1.00              2.00         1000.00
 holeyland             0.0000            0.00              1.20         1000.00
 wca1                  0.8000            0.50              1.00         1000.00
 wca2a                 0.8500            0.30              0.00         1000.00
 wca3a                 0.8000            0.50              0.00         1000.00
```

```
 wca3b                 0.8000           0.10             1.00         1000.00
 sa1                   0.7500           1.00             1.10         1000.00
 sa2                   1.0000           1.00             1.10         1000.00
 sa3                   0.3000           1.00             1.10         1000.00
 bcnp                  0.0000           0.00             0.00         1000.00
 enp                   1.0000           1.00             0.00         1000.00
 ocean                 0.0000           0.00             0.00            0.00 ;


param :        wsDemand        rainfall        runoff          evap   deltaStorage  :=
 /* unit           ac-ft         1E9ft^3         ac-ft       1E9ft^3         ac-ft  */
 lo                0.00          0.0000          0.00        0.1323      12165.28
 caloosa           0.00          0.0000       2166.00        0.0000          0.00
 stlucie           0.00          0.0000          0.00        0.0000          0.00
 miami             0.00          0.0000          0.00        0.0000       -748.36
 hillsnnr          0.00          0.0000          0.00        0.0000      -1245.45
 wpb               0.00          0.0000          0.00        0.0000       -538.91
 s4basin           0.00          0.0000          0.00        0.0000       -219.73
 s236basin         0.00          0.0000          0.00        0.0000        -30.82
 d298dist          0.00          0.0000          0.00        0.0000        -72.49
 brighton         79.70          0.0000          0.00        0.0000          0.00
 lops             69.10          0.0000          0.00        0.0000          0.00
 holeyland         0.00          0.0000          0.00        0.0128          0.00
 wca1              0.00          0.0000          0.00        0.0508        408.86
 wca2a             0.00          0.0000          0.00        0.0390      -1435.57
 wca3a             0.00          0.0000          0.00        0.1748       -709.34
 wca3b             0.00          0.0000          0.00        0.0363      -1923.88
 sa1               0.00          0.0000          0.00        0.0000          0.00
 sa2               3.51          0.0000          0.00        0.0000          0.00
 sa3              63.40          0.0000          0.00        0.0000          0.00
 bcnp              0.00          0.0000          0.00        0.0000          0.00
 enp               0.00          0.0000          0.00        0.0000          0.00
 ocean             0.00          0.0000          0.00        0.0000          0.00  ;

/*-------------------------------------------------------------*/
/* STRUCTURE parameters                                        */
/*-------------------------------------------------------------*/

 param :
 STRUCTURE :                name     designCap    maxFlow      manCon  :=
 /* unit                                 ft^3/s      ft^3/s      ft^3/s  */
         lo    caloosa        s77     20000.000   7800.000    6500.000
    caloosa         lo       s77b     20000.000   7800.000       0.000
    caloosa      ocean        s79     20000.000  12600.000   20000.000
         lo    stlucie       s308     20000.000   9136.066    2400.000
    stlucie         lo      s308b     20000.000  10000.000       0.000
    stlucie      ocean        s80     20000.000   8000.000   20000.000
   hillsnnr       wca1         s6     20000.000   2924.000   20000.000
   hillsnnr      ocean         s7     20000.000   2470.000   20000.000
      miami      ocean         s8     20000.000   4134.000   20000.000
         lo   hillsnnr         s2     20000.000   1512.000   20000.000
   hillsnnr         lo       s2bp     20000.000   1512.000   20000.000
         lo      miami       s354     20000.000   1512.000   20000.000
      miami         lo     s354bp     20000.000   1512.000   20000.000
         lo        wpb       hgs5     20000.000   1512.000   20000.000
        wpb         lo     hgs5bp     20000.000   1512.000       0.000
        wpb       wca1        s5a     20000.000   4790.000   20000.000
         lo    s4basin         s4     20000.000   1815.000   20000.000
    s4basin         lo       s4bp     20000.000   1815.000   20000.000
    s4basin      ocean       s235     20000.000    504.000   20000.000
         lo  s236basin       s236     20000.000    252.000   20000.000
  s236basin         lo     s236bp     20000.000    252.000   20000.000
         lo   d298dist       d298     20000.000    756.000   20000.000
   d298dist         lo     d298bp     20000.000    756.000   20000.000
  holeyland      ocean      hlcul     20000.000      0.000   20000.000
      miami  holeyland     hlpump     20000.000    750.000   20000.000
         lo  holeyland      lo2hl     20000.000   1512.000   20000.000
         lo   brighton brightonWS     20000.000    756.000   20000.000
```

```
        lo       lops     lopsWS     20000.000    756.000   20000.000
        lo       wca1    lo2wca1     20000.000   2931.206  100000.000
      wca1        sa1        s39     20000.000    504.159   20000.000
        lo      wca2a   lo2wca2a     20000.000   1037.115  100000.000
      wca1      wca2a        s10     20000.000      0.000       0.000
     wca2a        sa2     s34s38     20000.000   1555.354  100000.000
     wca2a      wca3a        s11     20000.000   2012.398  100000.000
        lo      wca3a   lo2wca3a     20000.000   1418.584  100000.000
     wca3a        enp    s12s333     20000.000   1557.070       0.000
     wca3a      wca3b       s151     20000.000   1500.000       0.000
     wca3b        sa3        s31     20000.000    851.109       0.000
     wca3a       bcnp       s343     20000.000    164.860       0.000
       sa1      ocean   tidalSA1     20000.000  10000.000   20000.000
       sa2      ocean   tidalSA2     20000.000  10000.000   20000.000
       sa3      ocean   tidalSA3     20000.000  10000.000   20000.000
       enp      ocean   tidalENP     20000.000  10000.000   20000.000
      bcnp      ocean  tidalBCNP     20000.000  10000.000   20000.000 ;


param :
STRUCTURE :                 name    useManCon   w_flow     minflow    :=
/* unit                                                   ft^3/s     */
        lo    caloosa        s77            1      0       0.000
   caloosa         lo       s77b            1      0       0.000
   caloosa      ocean        s79            0      0       0.000
        lo    stlucie       s308            1      0       0.000
   stlucie         lo      s308b            1      0       0.000
   stlucie      ocean        s80            0      0       0.000
  hillsnnr       wca1         s6            0      0       0.000
  hillsnnr      ocean         s7            0      0       0.000
     miami      ocean         s8            0      0       0.000
        lo   hillsnnr         s2            0      0       0.000
  hillsnnr         lo       s2bp            0      0       0.000
        lo     miami       s354            0      0       0.000
     miami         lo     s354bp            0      0       0.000
        lo        wpb       hgs5            0      0       0.000
       wpb         lo     hgs5bp            1      0       0.000
       wpb       wca1        s5a            0      0       0.000
        lo    s4basin         s4            0      0       0.000
   s4basin         lo       s4bp            0      0       0.000
   s4basin      ocean       s235            0      0       0.000
        lo  s236basin       s236            0      0       0.000
 s236basin         lo     s236bp            0      0       0.000
        lo   d298dist       d298            0      0       0.000
  d298dist         lo     d298bp            0      0       0.000
 holeyland      ocean      hlcul            0      0       0.000
     miami  holeyland     hlpump            0      0       0.000
        lo  holeyland      lo2hl            0      0       0.000
        lo   brighton brightonWS            0      0       0.000
        lo       lops     lopsWS            0      0       0.000
        lo       wca1    lo2wca1            0      0     241.850
      wca1        sa1        s39            0      0       0.000
        lo      wca2a   lo2wca2a            0      0     241.850
      wca1      wca2a        s10            1      0       0.000
     wca2a        sa2     s34s38            0      0       0.000
     wca2a      wca3a        s11            0      0       0.000
        lo      wca3a   lo2wca3a            0      0     241.850
     wca3a        enp    s12s333            0      0       0.000
     wca3a      wca3b       s151            0      0       0.000
     wca3b        sa3        s31            0      0       0.000
     wca3a       bcnp       s343            1      0       0.000
       sa1      ocean   tidalSA1            0      0       0.000
       sa2      ocean   tidalSA2            0      0       0.000
       sa3      ocean   tidalSA3            0      0       0.000
       enp      ocean   tidalENP            0      0       0.000
      bcnp      ocean  tidalBCNP            0      0       0.000 ;

end;
```

APPENDIX D

ADDITIONAL INFORMATION ABOUT FLOW CONSTRAINTS

This appendix contains further information about the management constraints and flow capacity constraints. The first section contains the code used to generate the management constraints in the LP model. The second section contains a sample of the code used to generate the flow capacity constraints in the LP model.

D.1. Management Constraints

This section contains the MATLAB code used to determine the management constraints for the LP model. The associated ORM constraints are implemented via *coordinators*; the information for each coordinator is contained in an XML input file. As mentioned above, each ORM coordinator regulates flow through a single structure for a particular purpose (either water supply or flood control) and therefore places a constraint only on part of the total flow. Elsewhere in the ORM code, the competing objectives are considered and the final flow is determined. This represents an important distinction between the current LP model and the ORM. This distinction prevented us from fully implementing all of the coordinators in the LP model. In Section 3.2 above, we discussed potential changes to the LP model that would allow for inclusion of additional management constraints.

Since the management constraints are not necessarily binding constraints in the ORM, we analyzed the ORM output in order to determine which constraints could be included in the LP model. Those constraints that were repeatedly violated in the ORM were omitted from the LP model; those constraints that were generally binding in the ORM were included in the LP model. Management constraints are included for six

structures in the LP model. The MATLAB code below determines the value of these management constraints. This code is contained in a MATLAB file called ManCon.m; this function is called by another MATLAB file (not shown) that generates the LP input file for each time step. The coordinators in ORM are contained in XML input files; it was necessary to modify that code in order to convert it into MATLAB code. In addition, not all of the management constraints for these six structures have been implemented. For example, the constraints on releases through structures S77 and S308 have been modified to allow for some flow at all times, in order to accommodate water supply needs. In the ORM, these constraints govern only flood control flow, so this consideration is unnecessary.

The following is the MATLAB code used to determine management constraints in the LP model:

```
function [manCon flowPen] = ManCon(structure,stage,ruleCurves,num1,num2,num3,num4)

% This function determines the value of the management constraint for six
% of the structures: S77, S308, S77bp, hgs5bp, S10, and S343.
% These management constraints are only enforced if the value of useManCon
% is set to 1 in generateGLPKinput.m

% The input arguments to this function are:
%   structure:  the structure for which the management constraint is being determined
%   stage:      generally the stage of the upstream basin
%   ruleCurves: vector containing the values of all rule curves for current period
%   num1, etc:  additional input data to be used in computing constraint

% The function outputs two values:
%   manCon:  this is the value of the management constraint
%   flowPen: penalty on flow through the structure; currently zero for all structures


rc6  = ruleCurves(1, 2);
rc8  = ruleCurves(1, 3);
rc9  = ruleCurves(1, 4);
rc10 = ruleCurves(1, 5);
rc11 = ruleCurves(1, 6);
rc13 = ruleCurves(1, 7);
rc14 = ruleCurves(1, 8);
rc15 = ruleCurves(1, 9);
rc16 = ruleCurves(1,10);
rc24 = ruleCurves(1,15);
rc25 = ruleCurves(1,16);
rc26 = ruleCurves(1,17);
rc27 = ruleCurves(1,18);
rc28 = ruleCurves(1,19);
```

```
flowPen = 0;

switch structure

    case ‘s77’ % flood control

        % leaving out lowest (zero flow) portion of step function
        % in order to accommodate water supply releases
        flowLevel1 = 6500; % pulse zone
        flowLevel2 = 4500; % Zone C release
        flowLevel3 = 6500; % Zone B release
        flowLevel4 = 7800; % Zone A release

        % check stage in Lake Okeechobee
        if (stage >= rc16)
            manConstraint = flowLevel4;
        elseif (stage >= rc15)
            manConstraint = flowLevel3;
        elseif (stage >= rc14)
            manConstraint = flowLevel2;
        else
            manConstraint = flowLevel1;
        end

        manCon = manConstraint;


    case ‘s308’ % flood control

        % leaving out lowest (zero flow) portion of step function
        % in order to accommodate water supply releases
        flowLevel1 = 2400; % pulse zone
        flowLevel2 = 2500; % Zone C release
        flowLevel3 = 3500; % Zone B release
        flowLevel4 = 8000; % Zone A release

        % check stage in Lake Okeechobee
        if (stage >= rc16)
            manConstraint = flowLevel4;
        elseif (stage >= rc15)
            manConstraint = flowLevel3;
        elseif (stage >= rc14)
            manConstraint = flowLevel2;
        else
            manConstraint = flowLevel1;
        end

        manCon = manConstraint;


    case ‘s77bp’ % flood control

        % check stage in Lake Okeechobee
        % no backpumping unless stage is less
        % than 10.5 ft
        if (stage < 10.5)
            manConstraint = 100000;
        else
            manConstraint = 0;
        end

        manCon = manConstraint;


    case ‘s308bp’ % flood control

        manConstraint = 0;
```

```
        % check stage in Lake Okeechobee
        % no backpumping unless stage is less than
        % 14 ft and less than rule curve 11
        if (stage < 14)
            if (stage < rc11)
                manConstraint = 100000;
            end
        end

        manCon = manConstraint;


    case 'hgs5bp' % flood control

        manConstraint = 0;

        % check stage in Lake Okeechobee
        % no backpumping unless stage is less
        % than 11.5 ft
        if (stage < 11.5)
            manConstraint = 100000;
        end

        manCon = manConstraint;


    case 's10' % flood control

        stage_wca1  = stage;  % upstream stage
        stage_wca2a = num1;
        stage_wca3a = num2;

        manConstraint = 100000;

        if (stage_wca1 < rc16)
            manConstraint=0;
        elseif (stage_wca2a >= rc9)
            manConstraint=0;
        elseif (stage_wca3a >= rc10)
            manConstraint=0;
        end

        manCon = manConstraint;


    case 's343' % flood control

        stage_wca3a = stage;  % upstream stage
        manConstraint = 100000;

        if (stage_wca3a < rc25)
            manConstraint = 0;
        end

        manCon = manConstraint;


    otherwise

        fprintf('ERROR in ManCon.m, with structure %5g  \n',structure);

end
```

D.2. Flow Capacity Constraints

This section contains sample code used to determine the maximum flow capacity constraints in the LP model. All of the capacity constraints from the ORM have been included in the LP model.[22] The only modification necessary was to convert the syntax to be compatible with MATLAB. The function shown below is contained in a file called MaxFlow.m; it is called by another MATLAB function every time period in order to generate the LP input file. In the interest of space, only part of the file is shown here, containing constraints for two structures.

```
function flowCapacity = MaxFlow(structure,givenHW,givenTW,pflow,value)

% This function determines the values of the flow capacity constraints.
% The information contained here was taken from the ORM source code.

switch structure
    case 'lo2wca1'
        hil = 500.0;
        %head = 0.0;
        upstrm_stg = givenHW;
        if (upstrm_stg < 12.0)
            hil = 0.0;
        end

        if (upstrm_stg < 12.0)
            head = 0.0;
        elseif (upstrm_stg < 12.5)
            head = upstrm_stg - 12.0;
        else
            head = upstrm_stg - 12.5;
        end

        wpb = 87.5*(upstrm_stg - 4.0)*(0.073*upstrm_stg - 0.168)*sqrt(head);
        Q1 = (hil+wpb);

        flowCapacity = Q1;


    case 'lo2wca2a'

        dnstrm_stg = givenTW;
        upstrm_stg = givenHW;
        %tailwater = 0.0;
        if (dnstrm_stg < 11.5)
            tailwater = dnstrm_stg;
        else
            tailwater = 48.03 * log10(dnstrm_stg) + 0.00071 * pflow ...
                        -.0009 * (dnstrm_stg^3.0) - 38.97;
        end

        headwater = min(upstrm_stg, 14.5);
        qmsq = 1423.07109 * (headwater^3.0) - 1331.58836 * (tailwater^3.0) ...
               + 179454.9430 * tailwater - 0.15511E7 * sqrt(headwater) ...
```

[22] In the ORM, the capacity constraints are located in the source code.

```
               + 3069176.95431;
        if (qmsq > 0)
            Q1 = sqrt(qmsq);
        else
            Q1 = 0;
        end
        %Q1 = (qmsq > 0) ? sqrt(qmsq) : 0.0;

        flowCapacity = Q1;


[... additional code omitted ...]


end  % end switch
```

REFERENCES

Boone, C. and Loucks, D. P. (2008). Development and assessment of an optimization component for the South Florida Regional Simulation Model. *Florida Watershed Journal*, 1(1), 8-11.

Dantzig, G. B. (1963). *Linear programming and extensions*. Princeton, NJ: Princeton University Press.

Lal, A. M. W., Van Zee, R., and Belnap, M. (2005). Case study: model to simulate flow in South Florida. *Journal of Hydraulic Engineering*, 131(4), 247-258.

Luenberger, D. G., and Ye, Y. (2008). *Linear and nonlinear programming* (3rd ed.). New York: Springer.

Park, J., Obeysekera, J., and Van Zee, R. (2007). Multilayer control hierarchy for water management decisions in integrated hydrologic simulation model. *Journal of Water Resources Planning and Management*, 133(2), 117-125.

South Florida Water Management District (SFWMD). (2005a). *Regional Simulation Model (RSM) Hydrologic Simulation Engine (HSE) user's manual*, Office of Modeling, Model Development Division (4540), West Palm Beach, Florida.

South Florida Water Management District (SFWMD). (2005b). *Regional Simulation Model (RSM) Management Simulation Engine (MSE) controllers: documentation*

and user manual, Office of Modeling, Model Development Division (4540), West Palm Beach, Florida.

South Florida Water Management District (SFWMD). (2005c). *Regional Simulation Model (RSM) Management Simulation Engine (MSE) supervisors: documentation and user manual*, Office of Modeling, Model Development Division (4540), West Palm Beach, Florida.

South Florida Water Management District (SFWMD). (2005d). *Regional Simulation Model (RSM) theory manual*, Office of Modeling, West Palm Beach, Florida.

Trimble, P. (1986). *South Florida Regional Routing Model,* South Florida Water Management District, Technical Publication 86-3, West Palm Beach, Florida.